Fall 2015

A Comparison of Alternative Interface Designs for Planning Software

Rachna Tiwary
San Jose State University

A COMPARISON OF ALTERNATIVE INTERFACE DESIGNS FOR PLANNING SOFTWARE

A Thesis

Presented to

The Faculty of the Graduate Program in Human Factors and Ergonomics

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Rachna Tiwary

December 2015

The Designated Thesis Committee Approves the Thesis Titled

A COMPARISON OF ALTERNATIVE INTERFACE DESIGNS FOR PLANNING SOFTWARE

by

Rachna Tiwary

APPROVED FOR THE DEPARTMENT OF

INDUSTRIAL AND SYSTEMS ENGINEERING

SAN JOSÉ STATE UNIVERSITY

December 2015

Dr. Kevin Jordan	Department of Psychology
Dr. Dorrit Billman	Senior Research Associate, NASA Ames Research Center
Dr. Sean Laraway	Department of Psychology

ABSTRACT

A COMPARISON OF ALTERNATIVE INTERFACE DESIGNS FOR PLANNING SOFTWARE

By Rachna Tiwary

The impact of varying the interface design of a software planning tool in a specific domain of the Attitude Determination and Control Operator (ADCO, where ADCO refers to a group or an individual) was studied. This study extended a prior study that compared two entirely different interfaces of a software planning tool, *LEGACY* and *NEW*, and by which it was found that the NEW interface better matched the underlying domain structure and resulted in improved performance. The current study looked into the impact of varying the levels of presentation of the elements of a plan on user performance in the NEW version of the software planning tool. The plan in the ADCO domain is a sequential grouping of events occurring in a temporal order across a timeline, regulating the movement and orientation of the International Space Station (ISS). Two elements of a plan, *Actions* and *Activities*, were organized within three levels, categorized as *Increment*, Activities, and Actions across the timeline. The two interfaces that were tested in this study were termed the *Hierarchical* version (delineating each of the three levels of elements, the Increment, the Activities, and the Actions) and the *Non-Hierarchical* version (presenting only the Increment and Actions). The study included twenty participants in all, with 10 each in the Hierarchical and Non-Hierarchical conditions. The results indicated that the users in Hierarchical version tended to perform better on certain task types than did users of the Non-Hierarchical version. However, the differences in performance on most tasks were not statistically significant. The key task that entailed the editing of events with the planning tool did not yield any differences in performance across the two versions.

ACKNOWLEDGEMENTS

I would like to thank my thesis committee members, Dr. Kevin Jordan, Dr. Dorrit Billman, and Dr. Sean Laraway for agreeing to work with me on my thesis and providing guidance throughout the completion of my thesis. I would like to thank the NASA Ames Human System Integration Division for its support and help throughout my tenure on base. My thanks to Dr. Michael Feary for his guidance and support, and Jessica Lee for her help in instrumenting the experimental materials. My special thanks to Dr. Dorrit Billman for her constant support and help with thesis writing.

I am deeply thankful to my family for their encouragement and patience through the process. Special thanks to my 6-year-old son, for his love each day that helps me evolve as a human.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Introduction

When designing interfaces for complex socio-technological work domains, one of the biggest challenges is to incorporate domain information into the design. Designers need to understand the relationship between all of the constituent elements of the domain and integrate and reflect the relationship in a meaningful way to the end users. The design challenge is to create a solution that adequately reflects the relationships of constituent elements while clearly communicating the scope and constraints to the end user. The current study focused on the specificities of information presentation for a software planning tool in a specific domain.

This study brought together the concept of organizing and representing information in a software planning tool designed to build and revise plans in a specific domain. The important design criterion is to understand the aspect of aligning the interface-design layout to match that of the underlying domain structure. An important goal of interface design is to identify the structural aspects of the domain that are most critical for the functional operation of software to help end users achieve their goals efficiently. The domain referred to in this study is the planning work of a niche group of users known as Attitude Determination and Control Operators (ADCOs). The ADCO could be a group or an individual. ADCO flight controllers are responsible for planning and controlling the orientation of the International Space Station (ISS). The product of the planning tasks is the ADCO plan document that is built and revised on the software planning tool.

A domain-structure analysis was conducted in an earlier experimental study that inspired the current study, which will be discussed in detail in a later section. The domain-structure analysis identified the structural organization of components of the ADCO plans and showed that these plans are composed of events organized hierarchically and aligned linearly across a

timeline. Broadly, the events were categorized into three elements, Increment, Activity and Action, with each element representing a different level of the hierarchy. An Increment is at the top level and an Action is at the lowest level of the 3-level hierarchy. The ADCO planning work motivated the current study that aimed to examine further the optimum level of information presentation in the context of a software planning tool that would enhance operator performance. To gauge the effect on user performance, the level of the hierarchy in the representation of elements of the ADCO was varied in the software planning tool, and the effect on users' perception of events and the plan as a whole was studied. The goal of this study was to understand the levels of detail that need to be incorporated in the interface design to facilitate optimum task performance.

Overview of Case Study Domain of ADCO

The domain in this experiment is the planning work of ADCO. ADCO functions include controlling the attitude (yaw, pitch, and roll) of the ISS. ADCO flight controllers are responsible for developing plans in advance of operations, as well as for monitoring the attitude of the ISS in real time. In addition, they also hold the responsibility of maintaining the flight attitude during quiet phases when attitude is not changing. It is during the quiet phases that the ISS is prepared for Activities in which an exchange of the crew and other resources takes place. To support these Activities, a sequence of Actions needs to occur in a specific temporal order.

A critical aspect of ADCO planning work is that the flight controllers are based at two different geographical locations: Houston, Texas, USA and Moscow, Russia. This international collaboration calls for detailed and advanced planning of the regulation of the ISS prior to the execution of any plan. The difference in geographical locations of ADCO flight controllers requires coordinated information exchange while the plan development is in progress between

the two groups. This makes the planning task on the software-planning tool critical as it calls for back-and-forth information exchange to facilitate appropriate regulation of the ISS's movement.

As mentioned above, the current study examined differences in user performance by varying the level of hierarchy of the elements' representation in the ADCO plans on the software-planning tool. The two versions of the software-planning tool that are compared in the study are categorized as Hierarchical and Non-Hierarchical. The current study tested the premise that the Hierarchical version of the software-planning tool would result in better performance compared to the Non-Hierarchical version. This study also tested the claim that an interface that better matches the hierarchy of the underlying domain structure will produce better performance.

The psychological importance of perceiving events by chunking them into a hierarchical representation has been demonstrated in various prior studies in different contexts, such as the recall and retrieval of list of words and the interpretation of an overall story narrative when only exposed to the scene level of a script (Bower, 1970). It has been supported widely in many earlier works of experimental psychology that information, when organized and presented into chunks with a systematic hierarchical sequence, is easier to understand and recollect (Bower, 1970; Zacks & Tversky, 2001). The insights of a few of the earlier works that illustrate the concept of information organization and its impact on perception, recall, and memory will be discussed below. These works that support the significance of hierarchy in information processing represent a small sample of the relevant research literature.

George Miller (1956), quoted by Paul m. Wortman (1975), in "Long-term Retention of Information as a Function of Its Organization", stated that by organizing the stimulus input successively into sequences of chunks, one manages to break the bottleneck of perceiving information. Bower (1970) studied the retrieval of words and concluded that a user's natural

optimal strategy was to categorize the semantic features of words into a broader, but smaller number of higher superordinate categories, thereby generating a hierarchy of nested subordinate sets. In his experiment, Bower found that participants who were presented with a sequence of words that aligned with the conceptual hierarchical order recalled about three times as many words as compared to participants who were presented with words randomly chosen from all the levels of conceptual hierarchy. In addition, in 90% of the cases, the participants demonstrated the top-down approach in hierarchical categorization in their recall, by recalling the superordinate word in the list earlier in the sequence than the subordinate in the experiment's protocol (Bower, 1970).

Further, Zacks, Tversky, & Iyer (2001) found that users have biases in perceiving ongoing activity in terms of discrete events organized hierarchically by a "partonomic" relationship. "Partonomy" in this context is a form of hierarchical relationship of events in which events can be viewed as organized into parts and sub-parts. For example, an automobile has parts like doors, windows, wheels, steering wheel, an engine, and seats. These parts can be further divided into sub-parts; for example, a seat generally consists of a seat base, backrest, armrest, seatbelt, and a headrest. In addition to partonomy, another form of hierarchy described by Zacks et al. that is applicable to event categorization is "Taxonomy." A taxonomical hierarchy categorizes events and defines the relation between events by "kind of" (in place of "part of") relationships (Zacks, et al., 2001). For example miniature golf is a kind of golf, which in turn is a kind of sport.

Baker and Wright (1954) quoted by Zacks (2001), found that most naturally occurring behavior is perceived by observers as partonomically organized (by part of relationships). This conclusion was based on extensive observation of children performing tasks in their daily lives.

Baker et al. found that close to 73% of behavior episodes of daily life's tasks partially coextended with other nearby episodes. Further, of these episodes, 90% were found to be partonomically related.

The importance of hierarchy in designing computer interfaces is further emphasized in Ecological Interface Design (EID) (Burns & Hajdukiewicz, 2004). EID approach, originated from the works of Vicente and Rasmussen (1989), largely focuses on designing interfaces for complex systems to help users efficiently visualize complex information relationships posed by the work domain. The EID approach to designing computer interfaces argues that to incorporate the domain information in design one needs to have a good understanding of the constraints of the work domain. Work Domain Analysis (WDA) is one approach that focuses on the constraints of the domain and environment of work. Research findings discussed in EID further support the notion that users perceive an interface as an integrated whole of objects, and the interface is created from many different levels of objects. These objects are nested in hierarchical form with each level nested under a higher parent level and all levels aligned along a dimension. The lowest level of objects in the hierarchy contains the individual pixels, and the highest level defines the upper level interface structure. The hierarchy of interface objects is segmented into different levels ranging from upper level workspace and view, to middle level graphic forms, to lower level graphic pixels. It is also reinforced by the EID approach that each of the higher levels are built on the levels below, and each higher level is dependent on the lower-level design decisions. The premise that hierarchical nesting of information in computer-interface design in complex work domains is critical in design decision supports the role of hierarchy in information design.

WDA consists of the *Abstraction Hierarchy* and the *Part-Whole Hierarchy*. The Abstraction Hierarchy is a post-design analysis of how the system works, and the Part-Whole Hierarchy disintegrates the system into subsystems and components (Burns & Hajdukiewicz, 2004). In the Abstraction Hierarchy, the connection between each level is defined by 'means-end' links. Further, "means-end" links are essentially "how/why" links, in which the level below explains how the level above is achieved. The Abstraction Hierarchy and Part-Whole Hierarchy together are instrumental in completing a Work Domain Model. The Abstraction Hierarchy describes various elements of the work domain, defining how closely the elements describe the physical nature of the work domain. The functional purpose and physical form are the two anchors of the Abstraction Hierarchy, in which the functional purpose of the domain is the highest level of abstraction in the hierarchy and the physical form describes the physical nature of the domain. The closer the element matches the physical nature of the domain, the less the degree of abstraction. The Abstraction Hierarchy puts forth five different levels, Functional Purpose, Abstract Function, Generalized Function, Physical Function, and Physical Form, to define a work domain. The Part-Whole Hierarchy is broadly based on the concept of aggregating components, in which levels are ordered one above another. The relationship between levels is defined by "contains" as one goes down the level and is part of as one moves up the level in the hierarchy. In contrast to the Abstraction Hierarchy, the Part-Whole Hierarchy has no set number of levels, and the number of levels is dependent on the complexity of the work domain and can be adjusted accordingly.

The hierarchy that is most relevant to the current study is the Part-Whole Hierarchy. For the ADCO work domain, the domain structure is specified by the ADCO plan, which is composed of events organized in a Part-Whole Hierarchy form that are aligned linearly across a

timeline. In this Part-Whole Hierarchical representation of the ADCO plan, the structural elements of the plan are broadly categorized as Increment, Activity, and Actions. Each of these three elements of the plan has a relative temporal ordering and an absolute time value associated with it. Increments are the largest element of planning that span across the time period between arrival and departure of the ISS crew, whereas Activity and Action are associated with durations smaller than the time-span range of the Increment. A detailed account of each of these structural elements is discussed in the following section. Broadly, the elements of the lower levels are defined by a part of relationship with the upper levels. In the ADCO domain, the Actions are nested as sub-elements of Activity, and Activities are further nested as parts of Increment. The elements are further organized along a horizontal axis to define the temporal order of their occurrence, aggregating to build up the final product of the domain, the ADCO plan.

In the current study, the two conditions, termed Hierarchical and Non-Hierarchical, varied in the visual delineation of one of the three structural elements of the ADCO plan, namely Activity. The Hierarchical Condition had explicit representation of all three structural elements of ADCO plan, whereas the Non-Hierarchical Condition only provided visual cues of Increment and Actions and had no explicit cue for Activity in the plan. Understanding the effect of chunking the events in the hierarchy on user performance in the ADCO domain was of key interest in the study.

Summary of the Prior Study in the ADCO Domain

A prior study (Billman, Arsintescu, Feary, Lee, Smith, & Tiwary, 2011) foundational to the current study compared two entirely different versions of the software planning tools designed to carry out the planning task of the ISS. The purpose of the prior study was to investigate the impact of matching the domain structure while designing a software interface for

greater efficiency. The study examined the effect on task performance of matching the software planning tool's design to the underlying domain structure. The two software planning tools in the prior study had very different interfaces and entirely different approaches to representing elements of a plan. The software planning tools in the prior study were termed NEW and LEGACY (Figure 1 and Figure 2 depict the two interfaces). The NEW interface had a continuous visual representation of the timeline over which the planning was done. NEW's layout explicitly delineated the component elements of a plan, whereas the LEGACY or old software planning tool was a text-based interface that provided no graphical representation of the elements of a plan. In LEGACY, the onus was on the user to decipher the relationships among the constituent elements of plan from long text files. In LEGACY, there was no explicit reference to the term Activity, and events were referred to as either a "Docking Mission" (Activity) or a "Docking Maneuver" (Action), leaving it to the user to interpret the relationship between the events. Furthermore, events were referred to as a file format, for example, "the UAF file", instead of the event itself. The study found that the NEW interface, which presented explicit delineation of the events of the plan, matched the underlying domain structure closely, and this resulted in faster and more accurate performance as compared to the LEGACY interface. The performance measures in that study were the speed and error rate (count and percentage of incorrect edits) while editing the elements of plan. This difference in the software planning tools design clearly affected the efficiency of performance of the participants in the prior study. The difference in the representation of the elements of plan in the prior study demonstrated the impact of the display interface on performance.

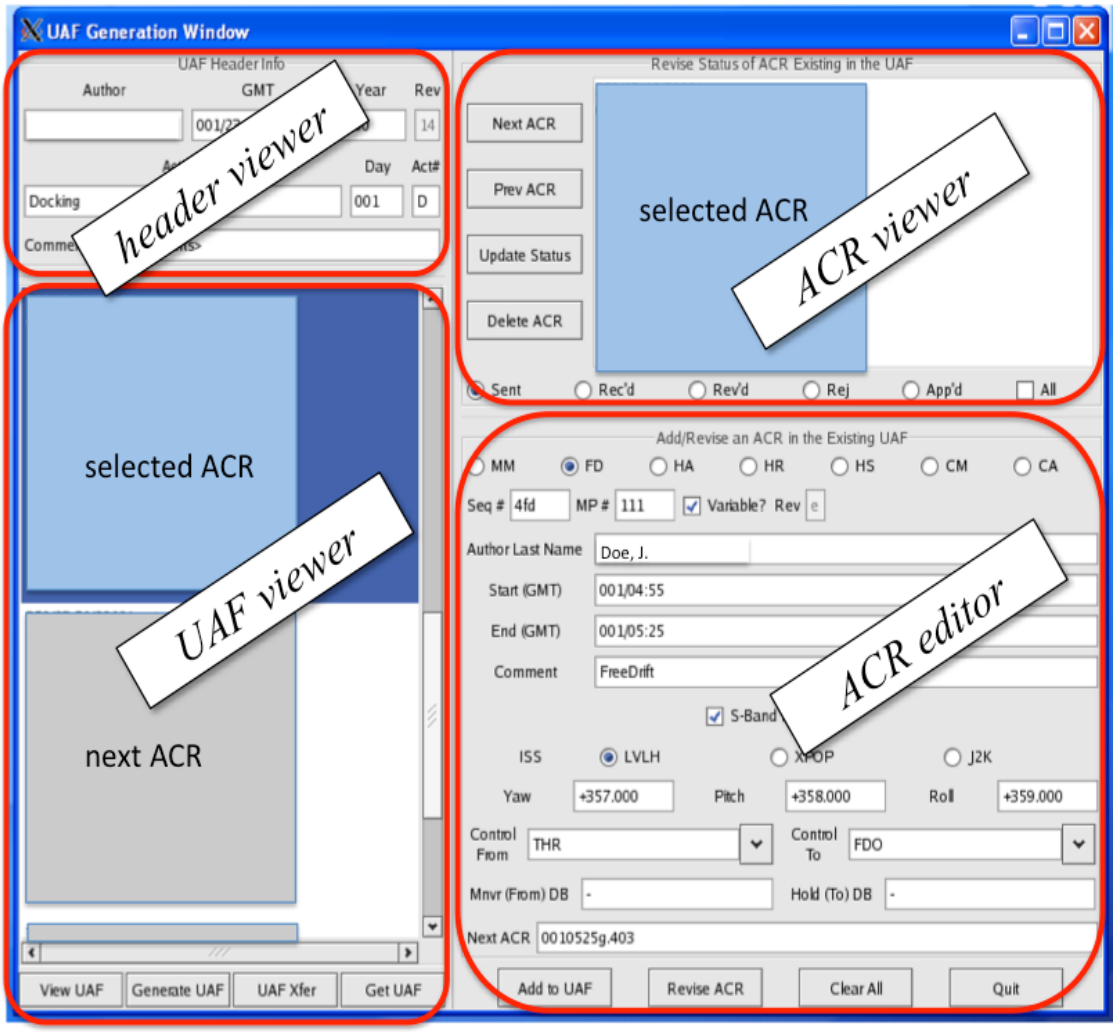

Figure 1. The interface of LEGACY software planning tool.

The interface has no explicit representation of Activity; the only visible element of a plan is the Actions. The four main functions panels for editing a plan are outlined in red. All names, values, and actions are invented as illustrations, and do not represent real values.

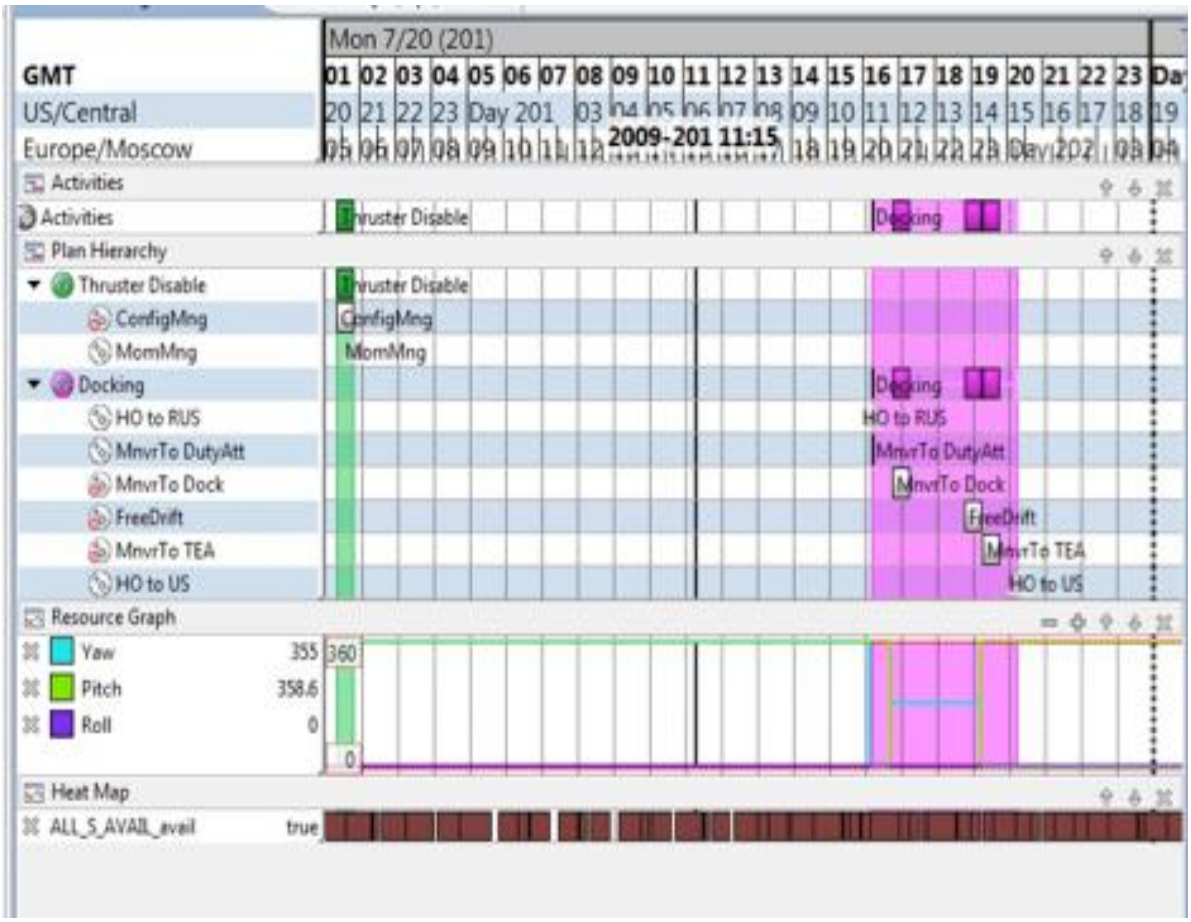

Figure 2. The interface of NEW software planning tool.

The interface of NEW representing elements of a plan such as Activities and Actions across timeline. The prior study suggested that it was important to understand and assess how explicit delineation of each level of the elements of a plan affected user performance when the user had a visual overview of the timeline of an ADCO plan. The current study sought to determine the effects of such delineation by comparing two representations of a plan, one with explicit delineation of Activity level and one without, on the NEW interface.

Motivation for the Current Study

The idea of the current study, as stated above, was conceptualized while working with the two software-planning tools in the aforementioned prior study. While working on the NEW

software planning tool, an intriguing problem was to understand the specifications of the hierarchy of the elements of a plan that would result in better user performance. The goal of the current study was to explore the hierarchy of representations of constituent elements of a plan. To do this, the levels of the plan-elements hierarchy were manipulated to get an insight into the impact of the hierarchical structure on user performance with the software planning tool. The elements of a plan were represented across a timeline. Each element's relationship with the other components in the hierarchy across the plan's timeline was visually represented in the software planning tool. The temporal nature of the plan was defined by three constituent elements, the Increment, Activity, and the Action. Figure 3 below illustrates the schematic representation of the hierarchical relationship of elements of a plan.

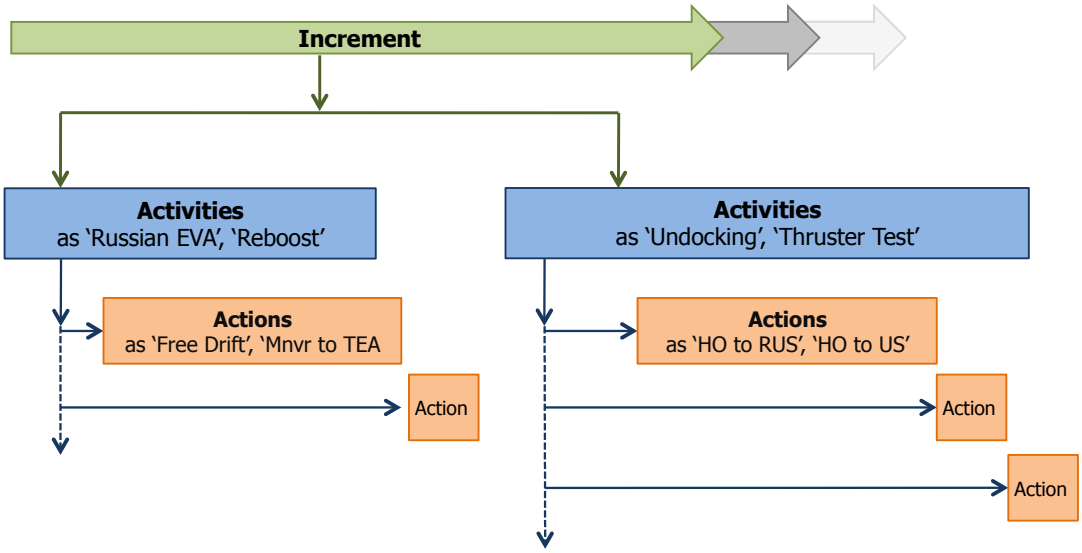

Figure 3. Schematic of structural construct of a plan.

Schematic is illustrating the structural relationship of elements of a plan where the elements are organized hierarchically into three levels-Increment, Activity, and Action. All elements were arranged in a relative temporal sequence and had absolute times associated with them. Each of these three elements was represented in discrete levels in the plans, which were representative of the hierarchy in a plan. The current study compared two versions of a software

planning tool that varied the visual representation of the elements of a plan. It tested whether the difference in information representation affected the user's performance on the software planning tool. A set of tasks was designed in the experiment for the Hierarchica*l* and Non-Hierarchical versions of the software planning tool. A detailed account of the differences in the two versions is discussed later with supporting schematics.

The three levels of hierarchy of a plan are the Increment, the Activity, and the Action. The Increment in a plan is signified by the time span between changes of the ISS crew. Each Increment constitutes of a range of specific Activities such as Docking, Undocking, Thruster Disable, Russian EVA, Relocate, Reboost, Thruster Test (US Master), and Thruster Test (RS Master). These Activities are planned by ADCOs. Each Activity comprises a specific sequence of Actions. These Actions are primarily maneuvers and handing over control (from USA to Russia and vice-versa) and are signified by specific attribute values. Figure 4 below depicts the relationship among events (Increment, Activity, and Action).

The Hierarchical version presented all three levels, the Increment, the Activity, and the Action. The Non-Hierarchical version of the tool had no representation of the Activity level. It only represented the Increment and the Action. Figure 4 and Figure 5 illustrated below are the schematic representation of the Hierarchical and Non-Hierarchical presentation of the elements of a plan.

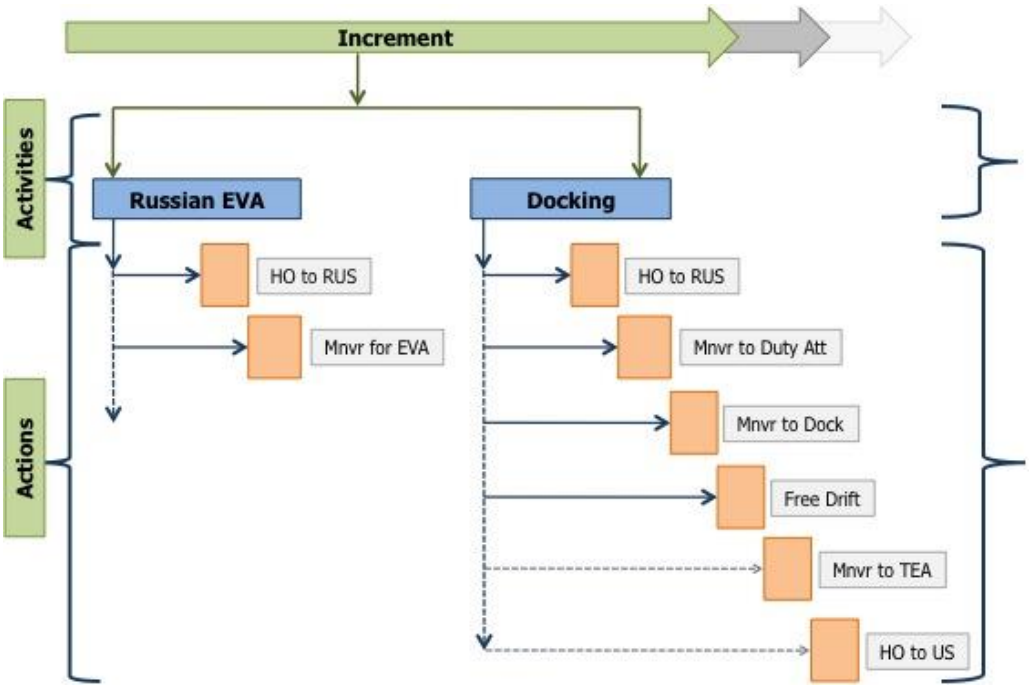

Figure 4. Schematic of Hierarchical condition showing Increment, Activities, and Actions.

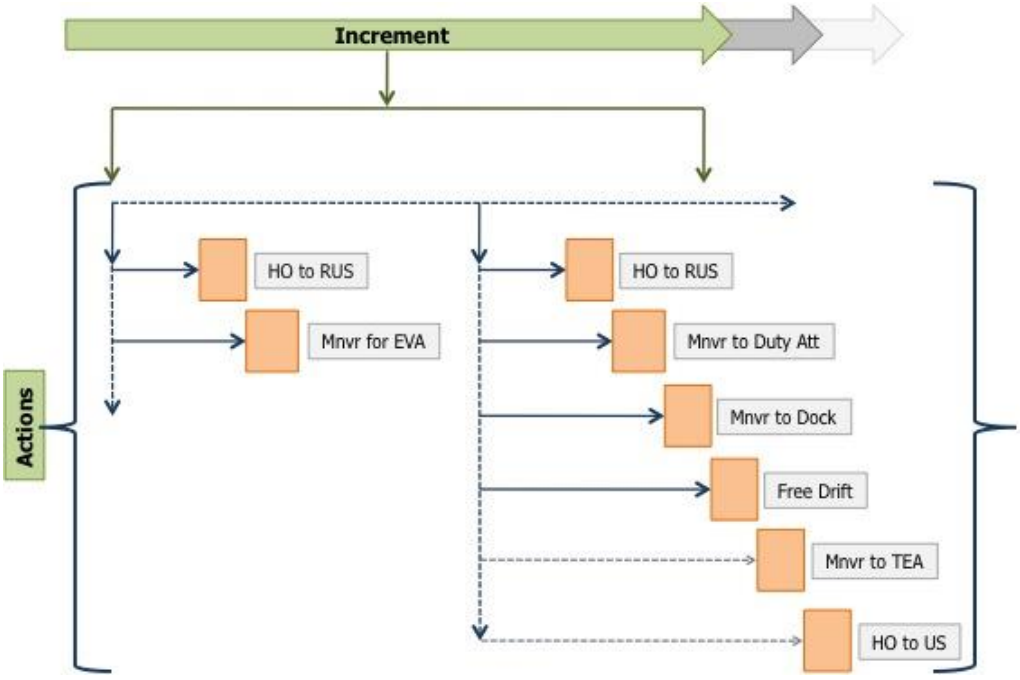

Figure 5. Schematic of Non-Hierarchical condition showing Increment, and Actions.

Overviews of the two software planning tool interfaces are illustrated in Figures 6 and 7 below. Figure 6 shows a screen shot of the Hierarchical version, and Figure 7 shows the Non-Hierarchical version of software planning tool used in the experiment.

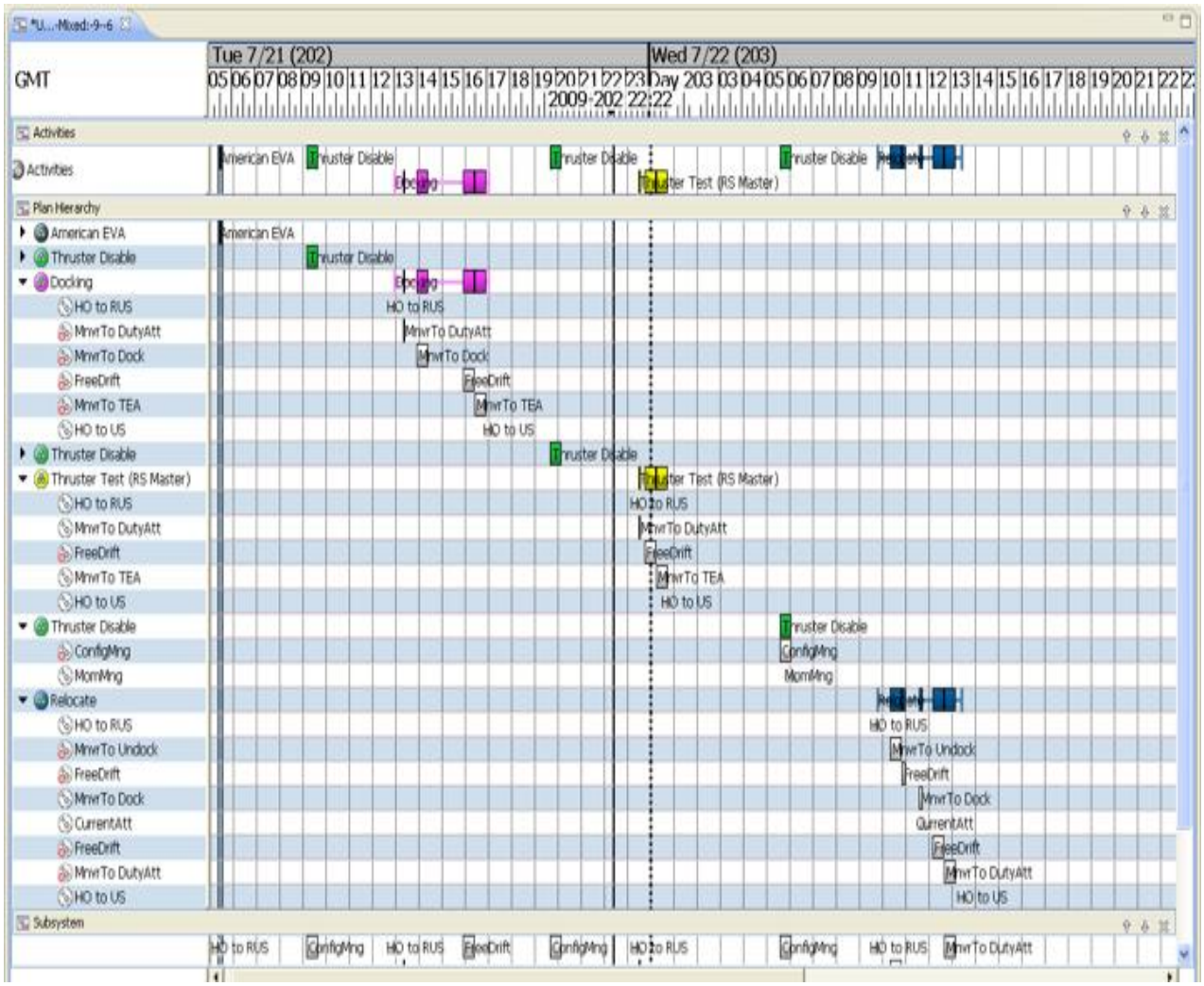

Figure 6. Screenshot of Hierarchical representation of software planning tool showing Activity, and Actions across timeline.

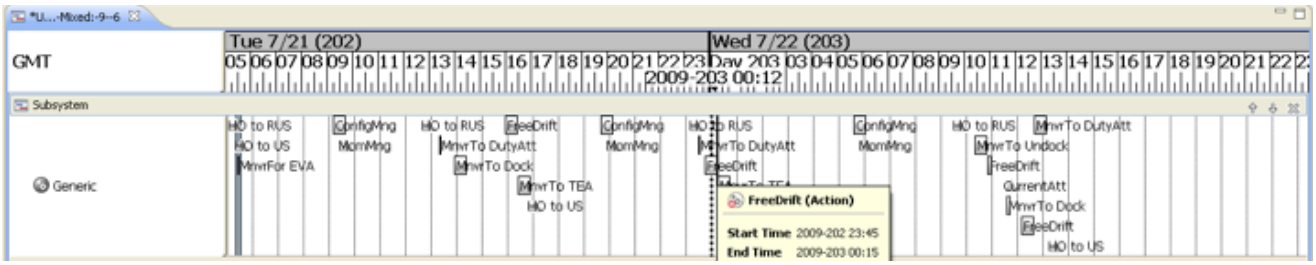

Figure 7. Screenshot of Non-Hierarchical representation of software planning tool showing Actions across timeline

Purpose of Study, Research Question, Prediction, and General Hypothesis

The primary goal of the study was to investigate the impact of information representation on task performance in a specific work domain that relies on a software planning tool. The measures of performance included speed and accuracy on assigned tasks. A secondary goal of the study was to investigate the effect of an interface's representation on a participant's conceptual understanding of the specific (ADCO) domain. Conceptual understanding of the ADCO domain in the current context refers to ability to correctly interpret the relationships of the constituent elements of a plan in the domain. The research objective was to compare user performance between the two versions of the software planning tool, Hierarchical and Non-Hierarchical, and understand whether the version type of the software planning tool affected the ability of the participant to perform the planning task.

It was predicted that the Hierarchical version of software that delineates all the three levels of elements of plans (Increment, Activity and Action) in the timeline would result in better overall performance compared to the Non-Hierarchical version of software that only represented two of the three levels of plan elements (Increment and Action). We therefore hypothesized that the Hierarchical version would result in higher accuracy (lower error-rate) and better efficiency (shorter Response Times) by the participants on the tasks that they were assigned to complete in the experiment. An additional hypothesis of the study was that the Hierarchical version, which

closely matches the underlying domain structure as compared to Non-Hierarchical, would facilitate better understanding and knowledge of the ADCO domain.

Method

Design

The independent variable of primary interest was the between-subject factor of the software-planning tool (Hierarchical versus Non-Hierarchical). In addition, for the Edit-Time Task, there were several Item Types, a within-subject factor. The Multiple-Choice Questions knowledge test was done twice, a within-subject factor for that task. The primary dependent variables were the response times (Speed) and errors (Accuracy) in tasks performed on the software planning tool.

In addition to the core tasks using the software-planning tool, there were additional conceptual tasks in the experiment, designed to get insights into the impact of domain training and usage of the software planning tool on participants' knowledge of the ADCO domain. A brief account of the dependent variables measured is as follows:

1. Speed: Response times on all item types in each task were recorded. Time taken to complete different sets of tasks was recorded in different file formats. For example, all the MATLAB generated times for Edit-Time Task were saved in a .xlsx file format at the end of the task. *TechSmith Morae Recorder* recorded the overall response time on various tasks. The .rdg format recording files generated by TechSmith Morae provided the best account of time taken to complete each task. In addition to this measure, a manual time-log was maintained for all participants across both Hierarchical and Non-Hierarchical Conditions. The manual time-log provided an approximate estimation of time on each task.

2. Accuracy: The number of tasks performed without error was also recorded. Each participant's responses were coded and scored to keep an account of correct responses for items in each task performed by the participants.

Participants

Twenty participants were included in the study, with 10 each in the Hierarchical and Non-Hierarchical Conditions. To minimize the impact of individual differences in the two software planning tool conditions (Hierarchical and Non-Hierarchical), all participants were recruited from upper-division undergraduate classes in the Aviation Department of San José State University. All participants were required to be permanent residents or citizens of the United States to facilitate their access to the NASA Ames Research Center. Participants did not have any prior experience or knowledge of the ADCO software planning tool used in regulating the ISS.

Participants were recruited using flyers at different locations at San José State University. Furthermore, to encourage aviation students to participate in the study, a brief talk about the scope of the study was also provided in an undergraduate class of SJSU's Aviation Engineering Department. Candidates who matched the above-mentioned criteria and expressed interest in participation were contacted via email and eventually over phone. Each participant was scheduled for 4 hours.

Experimental Setup and Apparatus

The study was conducted at the NASA Ames Research Center. Participants were tested individually in one of the two smaller rooms off of a larger common observation room. The experimenter was available throughout the session for assistance as needed. Three workstations were used for the study, two of which were used to run the experiment. The third workstation

was set up in the observation room to be used by the experimenter for observing the participant's computer screen, mirrored as the participant worked.

Each workstation consisted of a Samsung 32" Class LED 1080p 60Hz HDTV that served as monitor, a Lenovo keyboard (model KB1021), a Lenovo mouse (model MO28U0L), and a Lenovo CPU (model B3U.) Techsmith Morae *3.2* software was used to record the video files of all the visible screen events through the experiment. A group at NASA Ames developed the software planning tool on which the participants worked.

Procedure

At the beginning of the experiment, each participant was welcomed, briefed about the purpose of the study, and provided with consent form (Appendix B) to sign. After signing the consent form, the participants were provided with an overview of the tasks in the experiment. The tasks in the study were divided broadly into two categories, those performed with the software planning tool versus those performed without it, as listed in Table 1. The Edit-Time Task and the Build Plan Task were the two tasks that were performed on the software planning tool. ADCO domain training, a multiple choice questions, and the Error-Finding Task were performed without the software planning tool.

The software planning tool tasks were a close surrogate of the planning tasks that are performed by ADCOs for regulating the ISS. The planning tasks broadly entail revising, editing, and rescheduling events (Actions and Activities) across the timeline. The Edit-Time Task was the core task of the study as it was the closest match to the actual planning work. In this task, the participants were assigned to edit the time of events in a plan. The task had six item types, where each item type entailed editing the time of an individual event (an Action or Activity) or a group of events. There were four of each of these six item types, totaling 24 edit time items in

the task. A detailed account of this task type is provided in the section discussing this individual task type of Edit-Time Task on Software planning tool.

The responses to all the tasks that were performed off the software planning tool were saved in word file (.docx) format. These tasks were conceptualized and designed to assess participants' understanding of temporal relations of elements (Actions and Activities) in a plan.

Table 1

Two task Categories in the study, tasks performed with the software planning tool and task performed without the software planning tool

	Tasks performed without the software planning tool	Tasks performed with the software planning tool
Task 1	ADCO domain training: Text document based task. Participants noted their response in word (.docx) format file.	
Task 2	Multiple-choice questions First presentation: Task was MATLAB mediated.	
Task 3		Edit-Time Task: participants edited times of events of a provided plan. Task was MATLAB mediated.
Task 4		Build Plan Task: participants built plan from the template on the planning software.
Task 5	Multiple-choice questions Second presentation: Task was MATLAB mediated.	
Task 6	Error-Finding Task: participants looked for and reported the errors in the provided plan in word (.docx) format file	

ADCO domain training

To ensure that participants had a good understanding of the context of planning tasks of the ISS before working on the software, the study began with ADCO training. Participants learned about ADCO by reading a training document at the onset of the experiment. The document was organized into the following four sections:

- The first section provided a brief introduction to the ISS and its various aspects as a research laboratory, a habitat for continuous human occupation, a port for a spacecraft, and a vehicle in low Earth orbit.
- The second section outlined the basic structure and function of the International Space Station.
- The third section covered the act of controlling the ISS by regulating attitude (Yaw, Pitch, and Roll) during quiet phases and during activities.
- The fourth section described the ADCO's planning activities.

At the end of each section, participants were presented with a set of brief questions that required short, one-word to one-sentence responses. Responses were saved in word file (.docx) format.

The set of short questions at the end of each section served as knowledge checkpoints and an opportunity for the participants to get back to the experimenter in a timely manner with any questions that they might have had. Participants were instructed not to refer back to the training document while answering the questions. Precision in responses such as terminology and relevant details was not expected. Participants were assigned points if they demonstrated clarity in understanding and could explain their responses. The experimenter reviewed the responses

with the participant at the end of each section and answered any questions that the participant had.

Participants were informed at the onset of the experiment that the domain training should take an hour to an hour and fifteen minutes to complete. Each section was timed to range between 10-15 minutes. There was a 5-7 minutes time limit assigned to the task of answering the questionnaire at the end of each section. The experimenter assisted whenever the participant called for any assistance or at instances when the experimenter sensed the need to prompt the participant to increase the pace of reading the domain training.

Allocating participants to the two test conditions to have homogeneity across the two conditions (Hierarchical and Non-Hierarchical) was critical to the study. Participant performance on the domain-training questionnaire sections was used to assign the participants to one of the two conditions (Hierarchical and/or Non-Hierarchical). As participants answered the questions at the end of each section, the experimenter kept a log of their performance by observing their responses on Techsmith Morae Observer and scored each response. The cumulative scores on the questionnaire sections provided an insight of participant's understanding and learning of the ADCO domain. Participants were assigned to conditions to keep the scores roughly similar between conditions.

Multiple-choice questions-First Presentation

Multiple-choice questions were designed to assess whether working with the software planning tool affected the participants' understanding of the relationship of elements (Activity and Actions) of a plan. This task was presented twice, once before working with the software planning tool and once after the Build Plans Task.

Each participant was given 18 multiple-choice questions. The participants responded to each question presented on the computer before they could proceed to the next one. Once a response to the presented question was entered, the participant could not revisit the question or edit the response. Each question presented two possible sequences of constituent Actions of an Activity. One of the two options was a correct Actions sequence whereas the other option had scripted errors in it. The scripted errors belonged to one of the following types:

- Incorrect representation of point and interval Actions, for example presenting "HO to US" as an interval event although it is a point event or presenting "Maneuver to Thruster Test" as a point event while it is an interval event.
- Introducing extra Actions or eliminating an Action/set of Actions from an Activity.
- Flipping the sequence of Actions.
- Starting or ending the Action sequence within an Activity with an Action other than "Hand Over (to US or Russia)".

Participants clicked on the one of the two choices and MATLAB recorded the response time and scored whether the response was correct. Figure 8 below illustrates one of the questions of the Multiple-Choice Questions Task presented to the participant in MATLAB based graphic input box.

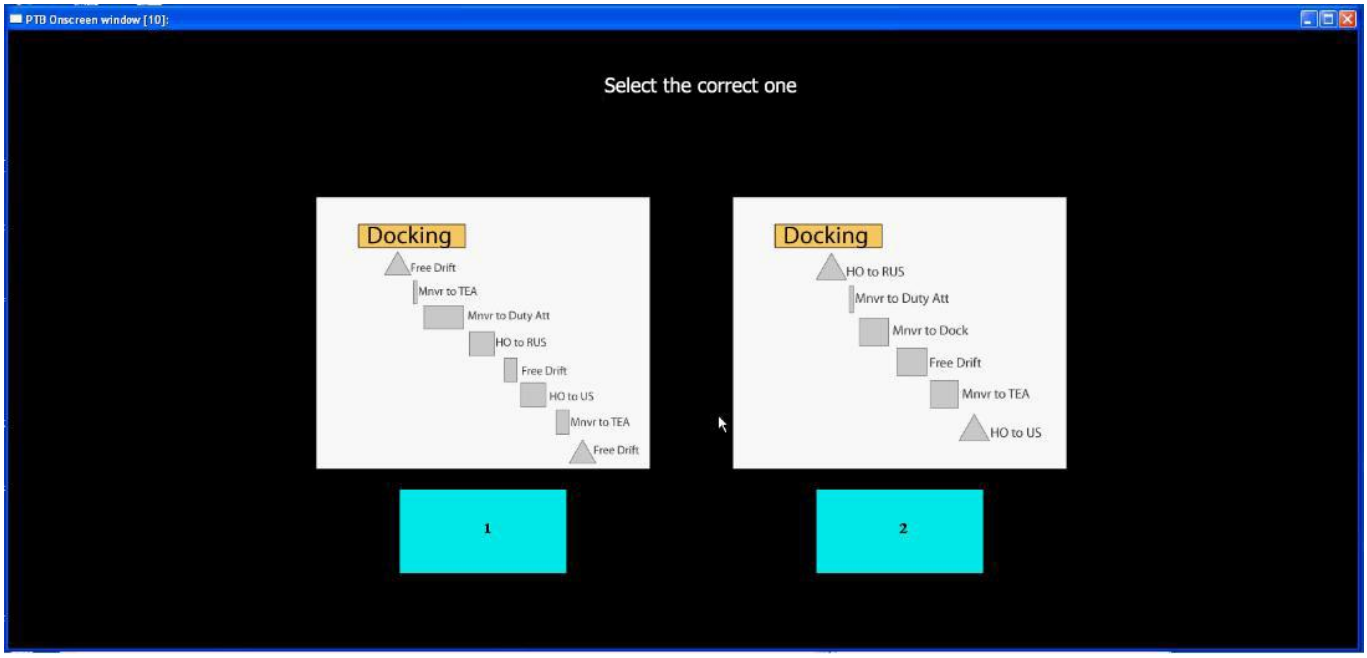

Figure 8. Screenshot of MATLAB generated graphic input box for Multiple-Choice Questions Task. In this example option 2 is correct

Software planning tool Training

Next, the participant was walked through the software planning tool training for the software version (Hierarchical or Non-Hierarchical) on which she or he was to work. The goal of the software planning tool training was to help the participant get acquainted with the functionality of the software planning tool and to acquire working knowledge of the software. A hypothetical and simple plan made up of two activities was presented for overview on the software planning tool. The experimenter walked the participant through different functionalities in the software. The experimenter followed a pre-designed script that delineated all the critical elements of a plan and provided an overview of functionalities for performing tasks using the software planning tool. On a broad level, the structure of each of the two training scripts was similar. Two versions of the training script were tailored to address the differences in presentation of elements of the plan in the two versions of the software planning tool in question.

After the demonstration of features of the software, the participant was provided with some simple tasks on the software planning tool to facilitate learning. At the end of the training, the participant was provided with an option to explore and work on the software planning tool for five to seven minutes to facilitate their familiarity with the workspace. The participant chose when to move ahead with the next task in the experiment.

Edit-Time Tasks

After the software planning tool training, the participants worked on the Edit-Time Task on the software. This was the key task of the study. This task was designed to gauge the differences in the participants' performance between the two versions of the software planning tool. The Edit-Time Task was designed to be similar to actual planning tasks performed by the ADCO.

In the Edit-Time Task, the participants who were assigned to work on the Hierarchical Condition of the software planning tool were able to see the visual representation of all three levels of plan, Increment, Activity, and Action. On the other hand, participants in the Non-Hierarchical Condition could only see the Increment and Actions of the plan. The primary challenge of the Edit-Time Task was to locate the correct set of Actions in question.

The Edit-Time Task had six item types. The first two item types entailed the editing of an individual event time, either an Action or an Activity. The remaining four item types involved shifting a group of actions, positioned *Adjacent* or *Non-Adjacent* to one another and grouped either *Within* one Activity or spanning *Between* two Activities. There were four of each item types, presented in blocks, each block consisting of six items (one of each type), summing to a total of 24 items in the task. The six item types, which were presented in each block in a

fixed order, were shifting time of an entire Activity (all Actions constituting an Activity), shifting time of an individual Action (of an Activity), shifting time of Adjacent Actions Within an Activity, shifting time of Non-Adjacent Actions Within an Activity, shifting time of Adjacent Actions spanning Between two Activities, and shifting time of Non-Adjacent Actions spanning Between two Activities.

The sequence of all six items in each of the four blocks remained the same through the Edit-Time Task. Understanding how users differed in their interaction with the two versions of the software planning tool across each of the six item-types in the Edit-Time Task was of principle interest to the current study. Another goal was to understand how each of the six item-types interacted with the two software planning tool conditions (Hierarchical versus Non-Hierarchical).

On a broad level, the item-types were categorized as an individual item (an Action and an Activity) or a group of items (Actions Within an Activity and Actions Between two Activities). The first two item-types were individual Activity and individual Action, whereas the next four item-types presented a group of actions. This group of actions were categorized as Adjacent versus Non-Adjacent, occurring either Within an Activity or Between two Activities. Some of the item-types included in this task do not occur in the real-time planning scenario in the ADCO domain; they were included to diagnose how differences in the interface might affect performance. Furthermore, an additional objective of categorizing the item-type was to provide the participant with diversity in the task type and to vary the level of difficulty, from the easiest being an individual Action, to the most challenging being a group of Actions across Activities. In addition to an overall prediction, additional predictions were made for the interaction of the item-types with the two software planning tool versions. The primary prediction of the effect of

the software planning tool conditions stated that participants in the Hierarchical Condition would have a shorter Response Time (RT) and higher accuracy than their counterparts in the Non-Hierarchical Condition in editing events. The following are the additional predictions, specific to the item-types that were also tested in the analysis of the Edit-Time Task responses:

- We predicted that the advantage of Hierarchical (shorter RT and higher accuracy) over Non-Hierarchical in editing individual Activity Items would be greater than the advantage of Hierarchical over Non-Hierarchical in editing individual Action Items.
- We predicted that overall the Within-Activity Items would be easier to edit than the Between-Activities Items for both conditions, but this difference would be greater in the Hierarchical Condition than in Non-Hierarchical Condition. Crossing an Activity boundary versus staying within an Activity might be an impediment in the Hierarchical Condition but of little consequence in the Non-Hierarchical Condition. In addition, there might be less or no visible difference across Within-Activity and Between-Activities Items for participants in the Non-Hierarchical Condition.
- Lastly for the Adjacent and Non-Adjacent Actions items, we predicted that overall, the Adjacent Actions items would be easier to edit than the Non-Adjacent Actions items. Furthermore, the participants in the Hierarchical Condition would have greater advantage over the Non-Hierarchical in editing Non-Adjacent Actions items than in editing Adjacent Action items. The explicit boundary of Activities would aid the more difficult task of selecting Non-Adjacent items for participants in the Hierarchical Condition.

For the Edit-Time Task, MATLAB-generated graphic input boxes were used to present each of the 24 items to the participants. The temporal order of the Actions was used as the cue to help the participant make the correct selection of events in question. An overview of the MATLAB-generated graphic input boxes used for Hierarchical or Non-Hierarchical Condition is presented below in Figure 9.

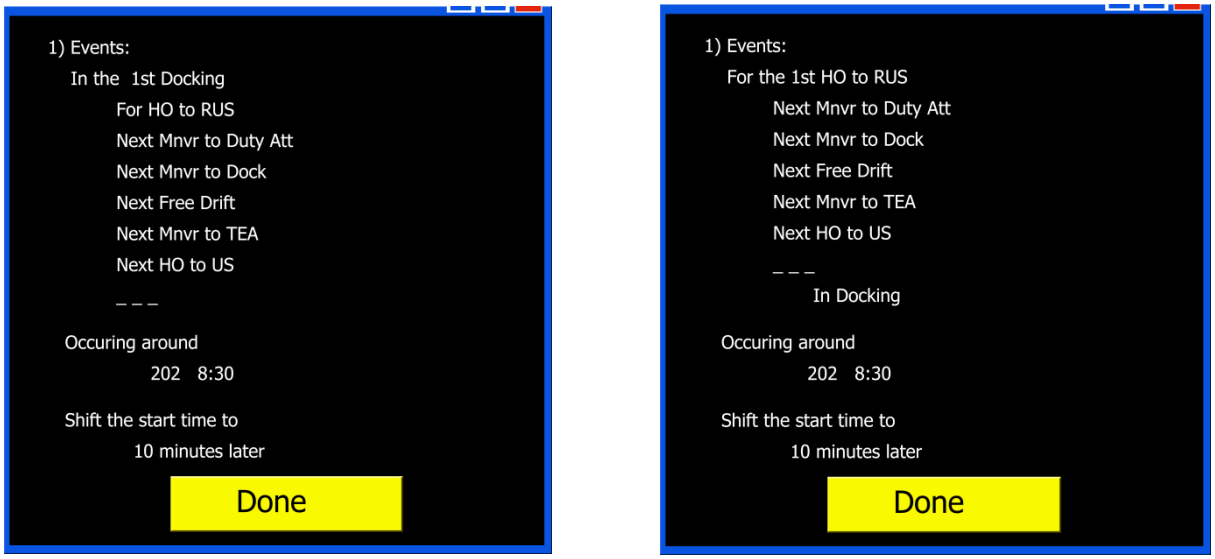

Figure 9. Screenshot of MATLAB generated graphic input box for Edit-Time Task.

In this example, the option on the left was for the Hierarchical condition and the option on the right was for the Non-Hierarchical Condition. The MATLAB graphic input box was designed taking into consideration the scope and limitation of the visual presentation of elements across the timeline in both the Hierarchical and Non-Hierarchical versions. For the Hierarchical Condition, the parent Activity was presented at the beginning as the first cue to locate the correct set of Actions in question, whereas for the Non-Hierarchical Condition, the first Action of a set of Actions served as the cue. The format in which MATLAB presented the Edit-Time Task was designed to support each Hierarchical and Non-Hierarchical version equally. Therefore, even

though there was no visual representation of parent Activity in the Non-Hierarchical version, the parent Activity was mentioned in the MATLAB graphic input box.

Build-Plan Task (Build Plan 1, Build Plan 2)

Next, the participants were provided with the task of building two plans from the set of templates present in the software planning tool. Building the plans from the template was one of the three conceptual tasks of the experiment that was aimed at understanding the effect of software usage on acquiring relevant domain knowledge. The plan built by each participant was representative of her/his understanding of co-relation and temporal order of Actions in an Activity and Activities in a plan. There were two performance metrics for this task. First was the time taken to build the plan and second metric took into account various aspects that contributed to the correctness of the plan. These aspects were correct sequence of Actions building up an Activity and spacing of events: Actions within Activity and Activities with respect to one another to closely match the temporal order of events in a plan.

The set of templates in the Hierarchical version differed from that of the Non-Hierarchical version in the Software planning tool. The templates in the Hierarchical version listed all the Activities (with all the relevant Actions nested underneath) required to build a plan. In contrast, the set of templates in the Non-Hierarchical version listed only the Actions of a plan (there was no cue of parent Activity listed for reference). We anticipated this would present a greater challenge to the participants in the Non-Hierarchical Condition, as they were required to show good knowledge of the Action sequence that would build up an Activity in addition to the Activity sequence that would make up a plan. Participants in the Hierarchical Condition could build a plan by simply having a good understanding of the Activity sequence and did not have to build the Actions needed for each Activity to build a plan.

Participants were given a document with instructions for building a plan that specified the activities to include and some timing constraints. Due to the difference in the template format across the Hierarchical and Non-Hierarchical Conditions of the Software planning tool, the instruction document for the participants in the Non-Hierarchical Condition was carefully scripted to address this disparity. The instruction document for both the Hierarchical and Non-Hierarchical conditions is provided in Appendix (see Appendices C and D). The document for the participants in the Non-Hierarchical Condition specified the sequence of all Actions that would make up the Activities to build the specific plan. In addition to this, the participants in the Non-Hierarchical Condition were also provided with an extra 10 minutes to build each plan compared to their counterparts in the Hierarchical Condition, as they were working at the Action level to build a plan.

Multiple-Choice Questions (Second presentation)

After the Build Plan Task, the participants were presented with the original Multiple-Choice Questions for the second time. The goal of presenting this task for the second time was to gauge the impact (if any) of the software planning tool (Hierarchical or Non-Hierarchical) on the participants' performance after the software planning tool usage.

Error-Finding Task

The last task of the experiment was to identify and report errors in a plan that had errors scripted into it. The Error-Finding Task was one of the two conceptual tasks that were performed without using the software planning tool. Participants were asked to identify the scripted errors and categorize the error type. Errors were carefully scripted into a plan made up of six Activities. The introduced errors belonged to one of the following categories:

- A missing Action in an Activity.

- An extra Action or set of Actions in a sequence.
- An incorrect Action name, for example ‘Maneuver for Thruster Disable’.
- An incorrect representation of point and interval events (point event such as ‘HO to RUS’ depicted as interval or interval event such as ‘Maneuver to TEA’ represented as point).

In all, there were 12 scripted errors in the provided plan. There were two instances of missing Actions (Maneuver to Duty Attitude and Maneuver to TEA in Reboost) and three cases of extra Actions (Free drift in Reboost, Maneuver to Duty Attitude in Russian EVA and Free Drift in Thruster Disable); there was one instance of incorrect Action name (Maneuver to Thruster Disable in Thruster Disable) and instances of incorrect representation of point and interval Actions; and there were two instances where the sequence of HO to RUS and HO to US was flipped. Participants were asked to identify these errors and to report their responses by noting them in a provided document (see Appendix E). The participant was informed of the range of the number of errors to expect in the provided plan at the onset of the task to facilitate the Error-Finding Task. A time limit of 15 minutes was associated with this task.

The metrics taken into account to gauge participants’ performance were the Response Time (RT) to identify the errors and the number of correctly identified errors. We predicted that participants in the Hierarchical Condition, who had a better visual experience of the levels of elements of plan, would demonstrate superior performance over users in the Non-Hierarchical Condition in reporting the scripted errors. This prediction was made for both of the performance metrics, the RT, and the total number of correctly identified errors.

Debriefing

The debriefing session started with structured questions where the participants were asked about their experiences as likes and dislikes of the aspects of the software planning tool (see Appendix F). Participants were encouraged to share their positive and negative experiences while working on the software. They were encouraged to speak about the challenges they faced and were asked to voice what could have made their experience better. In addition, the participants were also encouraged to share any ideas they might have had for the software planning tool that could have helped them perform better. This was experimenter-mediated interaction, and the participant's responses were noted by the experimenter and saved in word file (.docx) format.

Results

We predicted that overall, the participants in the Hierarchical Condition would work faster and have fewer errors than their counterparts in the Non-Hierarchical Condition across each of the task types: the Edit-Time Task, the Build-Plan Task, the multiple-choice questions, and the Error-Finding Task. We made predictions of the main effect of the primary independent variable, the software planning tool version, and the additional independent variable (task type) on performance metrics as RT and error count, for each task of the experiment. Analyses were also done to study the interactions of the independent variables. The results for each of the tasks were analyzed individually, as each task was designed and presented differently. For example, unlike all the other tasks that were designed only to compare user performance between conditions (Hierarchical and Non-Hierarchical), the multiple-choice questions also assessed learning and were presented twice to the user within each Hierarchical and Non-Hierarchical Condition, once before working with the software planning tool and once after working with the software planning tool.

Most of the data were analyzed using SPSS software with a mixed measure ANOVA and *t* tests for independent samples. For each of the tasks, the software planning tool version (Hierarchical versus Non-Hierarchical) was the primary independent variable. Additional independent variables concerning item types were examined depending on the task-type.

Edit-Time Task

For the Edit-Time Task, it was predicted that participants in the Hierarchical Condition would have an overall advantage of a faster RT and higher accuracy (fewer errors) than would their counterparts in the Non-Hierarchical Condition in editing the events in question.

In the Edit-time task for the Hierarchical condition, there were instances of three performance data points that were much lower than the average of the 10 participants across both of the software planning tool conditions. These three users in Hierarchical condition never operated at Action level and only operated at Activity level, resulting in 5 incorrect responses of the 6 item types. This resulted in only 4 overall correct responses of the 24 tasks in all in the Edit-time task category. Therefore, these three outliers in the Hierarchical condition were not included in statistical analyses of Edit-time task.

A 2 (Software planning tool) x 6 (item-type) mixed measures ANOVA was conducted to determine the effect of the Software planning tool Conditions, Hierarchical [M = 16.3, SD = 2.31] versus Non-Hierarchical [M = 18.3, SD = 2.74], on RT. RT of correct responses were included in the statistical analysis. The analysis indicated a non-significant effect of the Software planning tool Conditions on RT (of correct responses), $F(1,15) = 1.19$, $p = .293$, $d = .824$, Greenhouse-Geisser adjusted. RT are shown in Figure 10.

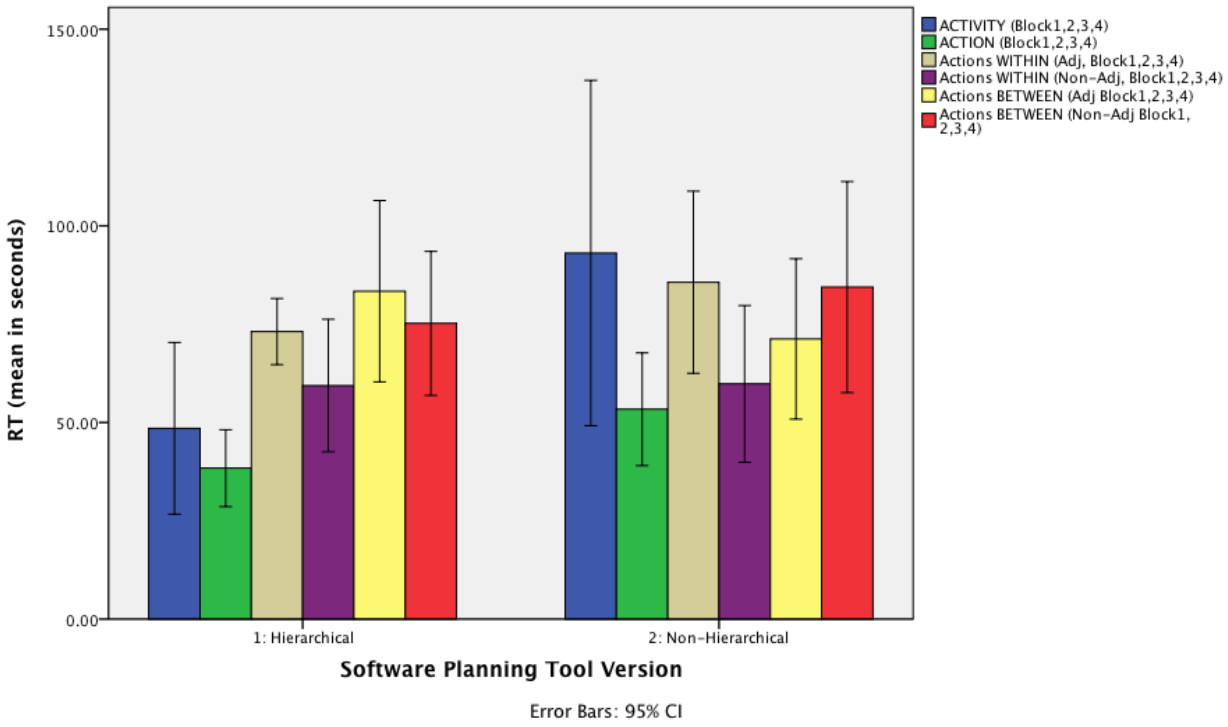

Figure 10. RT of correct responses for Edit-Time Task across two versions of the planning tool for the six item-types.

In summary, the difference in RT for correct responses between the two software planning tool conditions was not large enough to reach statistical significance. The difference in RT (for correct responses) was most noticeable for the individual Activity item-type of all the six item-types in the task, where participants in the Non-Hierarchical condition took longer than participants in the Hierarchical condition. This was further tested for statistical significance and the findings are presented in the next section.

The second prediction for the Edit-Time Task stated that participants in the Hierarchical Condition would have shorter RT in editing an individual Activity than an individual Action. A 2 (Software planning tool) x 2 (item-type: Activity versus Action) mixed measures ANOVA was used to determine if there were any significant differences in RT of editing the two item-types (individual Activity and individual Action) across two Software planning tool Conditions. Results yielded no significant main effect of the Software planning tool Conditions on RT (for

correct responses), F (1,15) = 4.47, p = .052 (see Figure 11), and no significant main effect of the two item-types on RT (for correct responses), F (1,15) = 4.51, p = .051. Further, the ANOVA yielded no interaction effect of the Software planning tool Conditions and the two item-types on RT [F (1,15) = 1.59, p = .227].

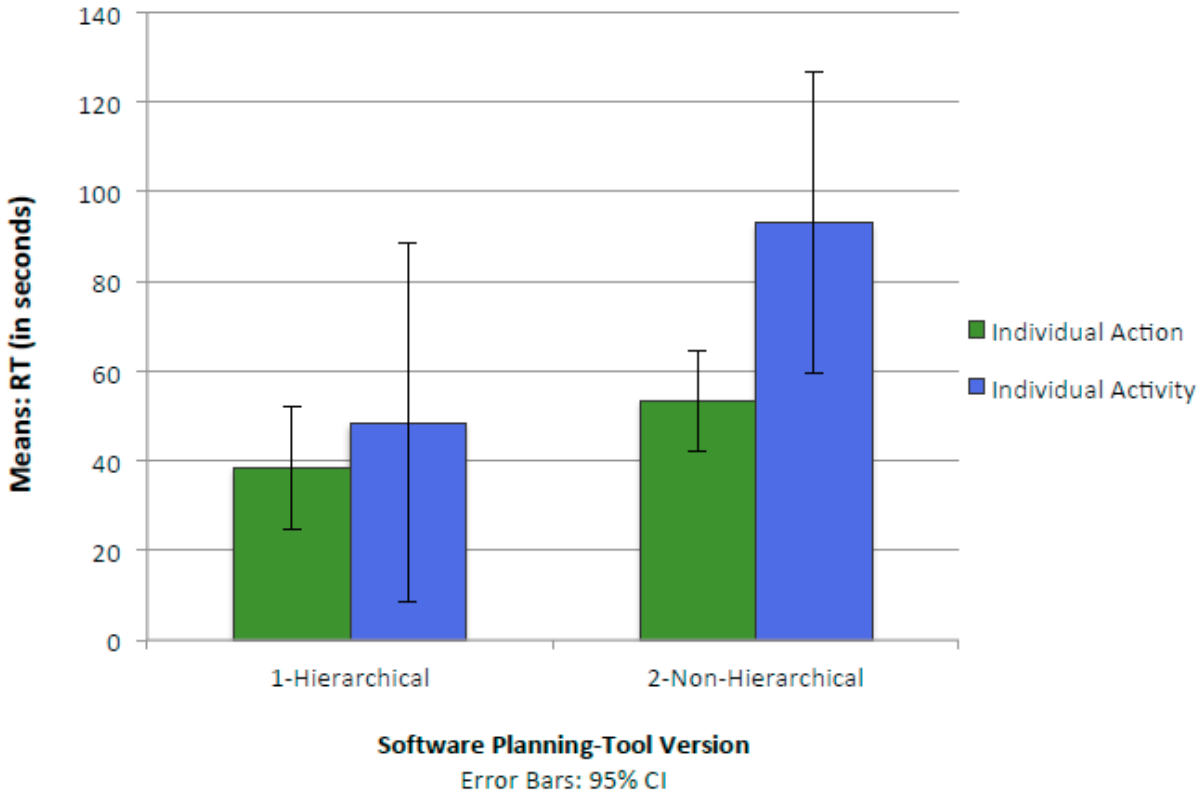

Figure 11. RT of correct responses for Edit-Time Task across two versions of the planning tool for the two item-types: Individual Activity and Individual Action.

Third, we predicted an interaction of Software planning tool with Within Activity versus Between Activity item types. This prediction suggested that participants in the Hierarchical Condition would have a shorter RT and fewer errors in editing actions grouped Within Activity than those grouped Between Activities. This prediction further suggested that in the Hierarchical Condition the participants would perform substantially better for Within Activity item-type than Between Activities item type (faster RT and fewer errors) while in Non-Hierarchical these item-

types would not differ. These predictions were based on the premise that the delineation of Activities and Actions group was explicit for participants in the Hierarchical Condition; hence, it would be easier for them to select and edit a group of Actions within an Activity compared to Actions spread across two different Activities.

A 2 (Software planning tool) x 2 (Item-Type: Within versus Between) mixed measures ANOVA was performed to study the effect of the Software planning tool Conditions (Hierarchical and Non-Hierarchical) on RT for correct responses. The ANOVA yielded no significant effect of the Software planning tool Conditions on RT (for correct responses), $F(1,15) = 0.054$, $p = .819$, Greenhouse-Geisser adjusted (see Figure 12), and no significant effect of action grouping (Within or Between Activities) on RT, $F(1,15) = 3.70$, $p = .074$, Greenhouse-Geisser adjusted. Further, the analyses yielded no significant interaction effect of two IVs (Software planning tool Conditions and Item-type: Within versus Between) on RT $F(1,15) = 0.71$, $p = .413$.

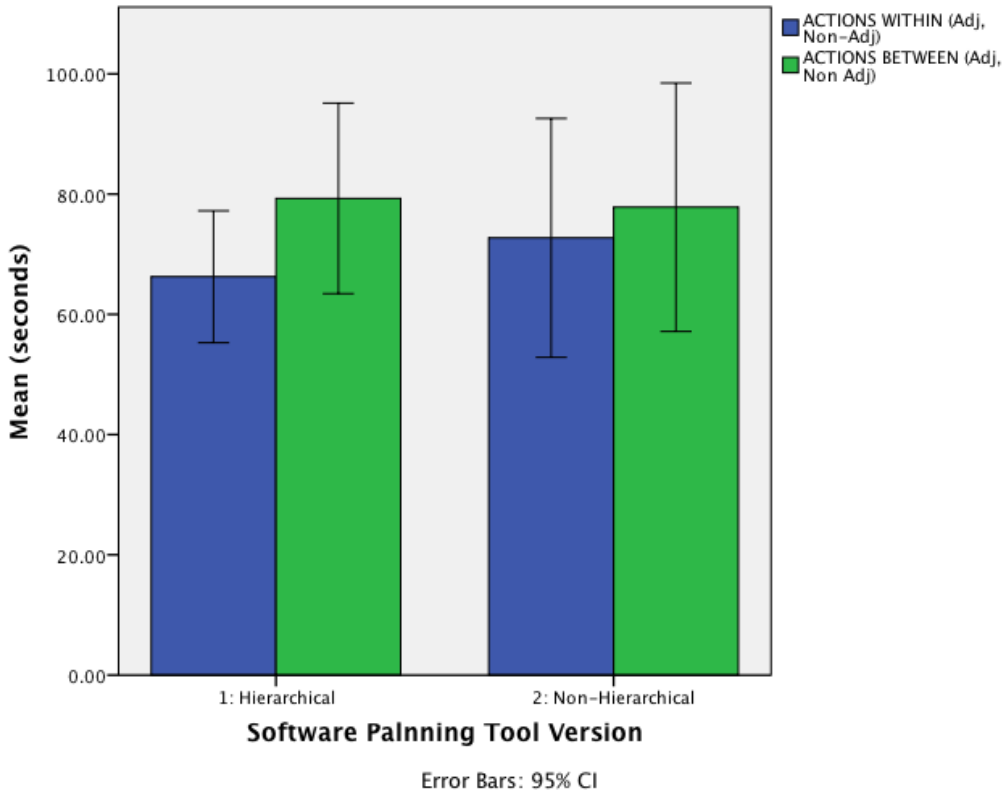

Figure 12. RT of correct responses for Edit-Time Task across two versions of the planning tool for two item-types of Action grouping: Within and Between Activity.

Fourth, we predicted that participants using the Hierarchical interface would have shorter RT and fewer errors than the participants using the Non-Hierarchical interface for editing Non-Adjacent events. We also predicted that Adjacent Action items would be easier to edit than the Non-Adjacent Action items in both the Software planning tools. A 2 (Software planning tool) x 2 (Item-Type: Adjacent versus Non-Adjacent) mixed measures ANOVA was performed to study the effects of the Software planning tool Conditions and the actions' spacing (Adjacent and Non-Adjacent) on RT. The analyses took the RT for correct responses into account. The ANOVA yielded no significant effect of the Software planning tool conditions and the Action distribution on RT, F (1,15) = 0.054, p = .819, (see Figure 13). No significant effect of Action spacing (Adjacent versus Non-Adjacent) was found on mean RT for correct responses, F (1,15) = 3.58, p = .078. Lastly, no statistically significant interaction of the Software planning tool Conditions

and Actions' spacing (Adjacent and Non-Adjacent) was found on the RT, F (1,15) = 0.258, p = 0.619.

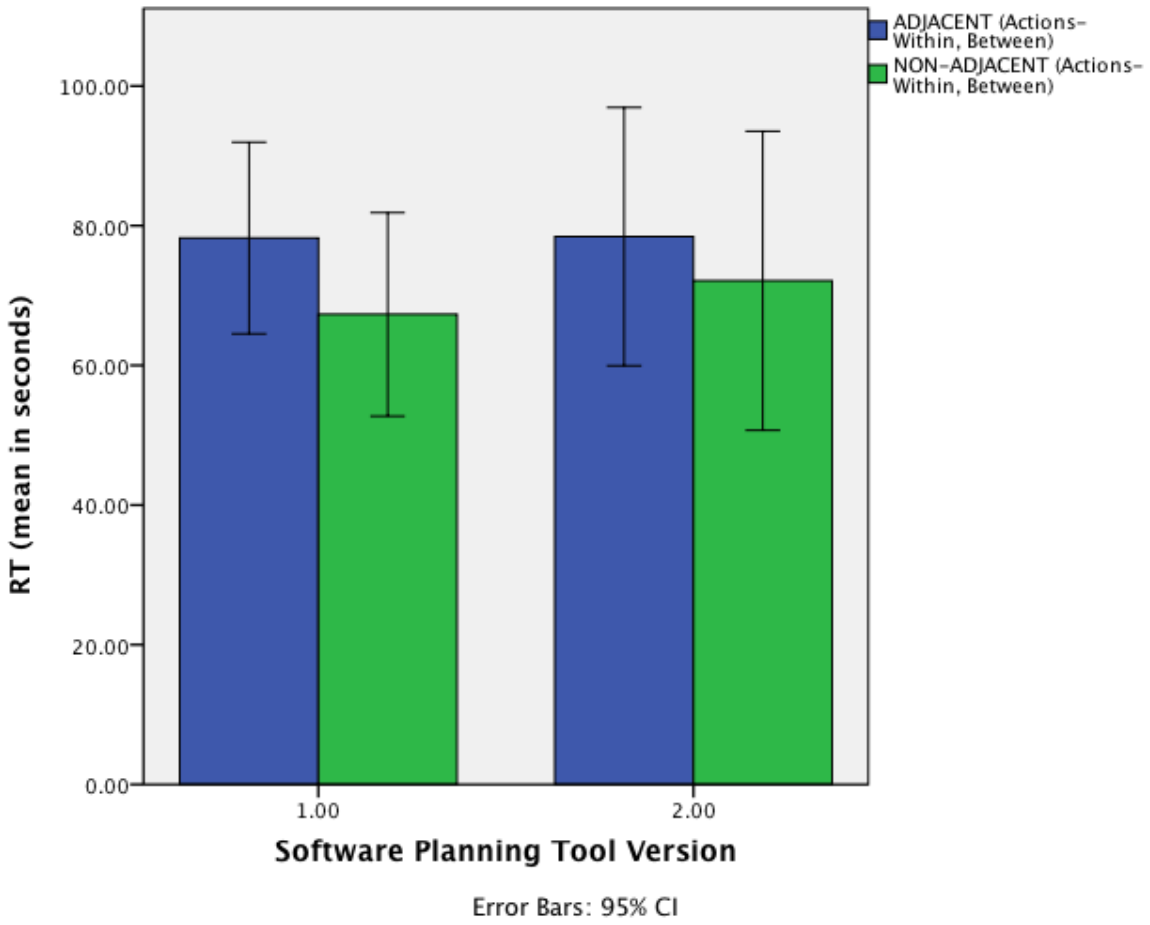

Figure 13. RT of correct responses for Edit-Time Task across two Software planning tool versions for two item-types of Actions spacing: Adjacent and Non-Adjacent.

Multiple-Choice Questions

The two independent variables for the Multiple-Choice Questions task were the two Software planning tool Conditions, and the Before-After Tool Usage Conditions. It was predicted that users would have shorter RT and fewer number of errors within both the Hierarchical and the Non-Hierarchical conditions after having used the Software planning tool than before the Software planning tool usage. This was based on the assumption that after having experience with editing the events of a plan with the Software planning tool, users would

have a better understanding of the relationship among elements in a plan. In addition, it was also predicted that participants in the Hierarchical Condition would have shorter RT and fewer errors than their counterparts in the Non-Hierarchical Condition while answering the Multiple-Choice Questions after they have worked on the Software planning tool.

A 2 (Software planning tool) x 2 (Before-After Usage) mixed measures ANOVA was performed on RT and on correct responses. The ANOVA on RT yielded no significant effect of the Software planning tool Condition, F (1,18) = 0.533, p = .475, (see Figure 14), a significant effect of Before-After Usage Condition on RT, F (1,18) = 38.56, p < .001 and no significant interaction was found, F (1,18) = .002, p = .964. The ANOVA on correct responses found no significant main effect of Software planning tool Condition, F (1,18) = 0.005, p = 0.945, (see Figure 15), a significant main effect of Before-After Usage Condition, F (1,18) = 14.38, p = .001, and no significant interaction, F (1,18) = .31, p = .583.

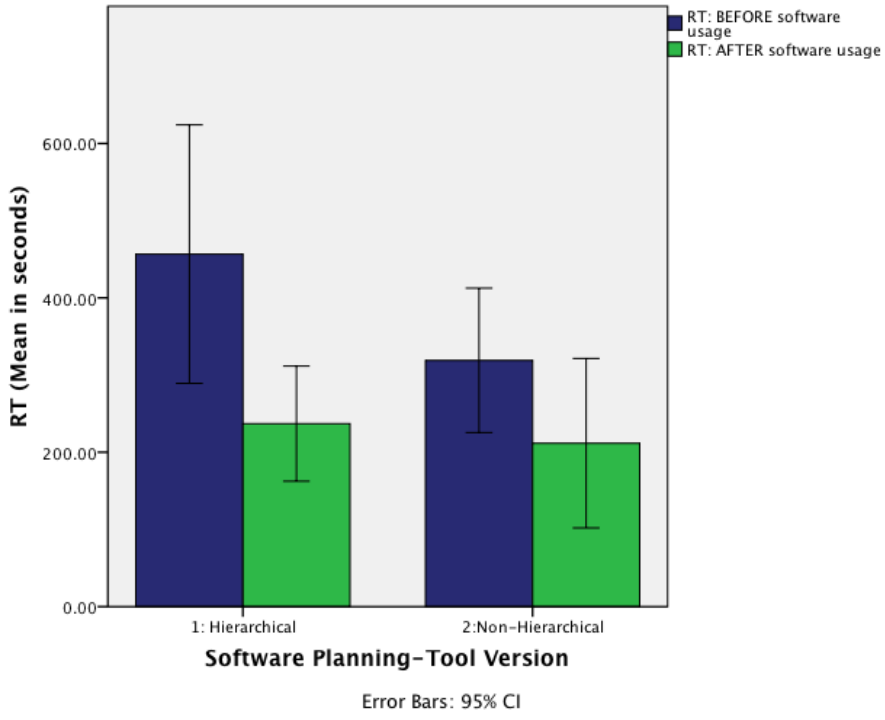

Figure 14. RT of Multiple-Choice Questions, across two versions of pre and post software planning tool usage condition.

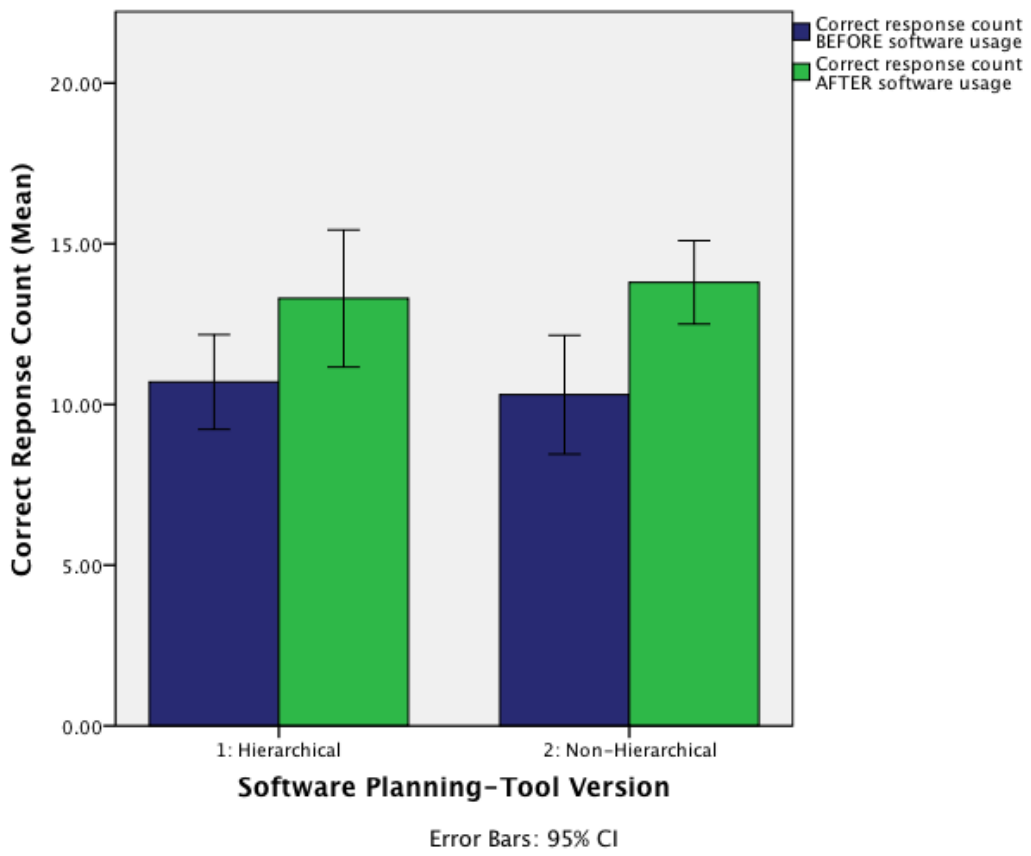

Figure 15. Correct response count for multiple-choice questions, across two versions of pre and post software planning tool usage condition.

Although RT was not significantly different across the two Software planning tool Conditions, the participants in Hierarchical Condition, who tended to be slightly slower initially prior to Software planning tool usage, tended to speed up more from the first to the second test (Figure 14). The participants in the Hierarchical Condition tended to take longer than the participants in the Non-Hierarchical Condition.

Furthermore, for the correct response count metric, the data did indicate improved performance by participants across both the conditions (Hierarchical and Non-Hierarchical) after having worked on the Software planning tool, this advantage was found to be statistically significant. Participants in the two conditions (Hierarchical and Non-Hierarchical) scored very close to each other in reporting errors for both Before and After Software planning tool usage

conditions. Contrary to the prediction, participants in the Non-Hierarchical Condition show slightly more improvement compared to their counterparts in the Hierarchical Condition moving from the first test to the second test.

Build Plan Task

For each of the two plans that were built, the time taken to build the plan was one of the two performance metrics. The other metric looked into various aspects that together contributed to the overall correctness of the built plan. We scored the built plans for correct inclusion of the relevant Activities and Actions in question, correct placing of the Activities and Actions (within Activity) in sequential order, and spacing Activities and Actions with respect to each other as directed to complete a plan. Plans were scored for including correct components and spacing events (Activities and Actions) to meet two or three criteria given in the problem specification. An error was scored for every required event that was missing or any incorrect event that was included. Similarly, an error was scored for incorrect spacing of an Activity or Action if the spacing deviated from a specified requirement.

For the Build Plan Task, we predicted that users in the Hierarchical Condition would have shorter RT and would build more accurate plans (with fewer errors) than their counterparts in the Non-Hierarchical Condition. An independent sample *t* test was performed to study the effect of the Software planning tool Conditions (Hierarchical versus Non-Hierarchical) on the performance measures. There was a statistically significant difference in RT for Hierarchical (M = 544.85, SD = 197.60) and Non-Hierarchical (M = 963.33, SD = 222.71) Software planning tool Conditions, t (17) = -4.34, $p < .001$. Figure 16 illustrates this significant difference in the mean of Response Time for the Build Plan Task for the two Software planning tool Conditions. However, no statistically significant effect of Software planning tools was found on the overall

error count across the two plans, t (17) = -0.35, p = 0.73. Figure 17 below delineates this finding.

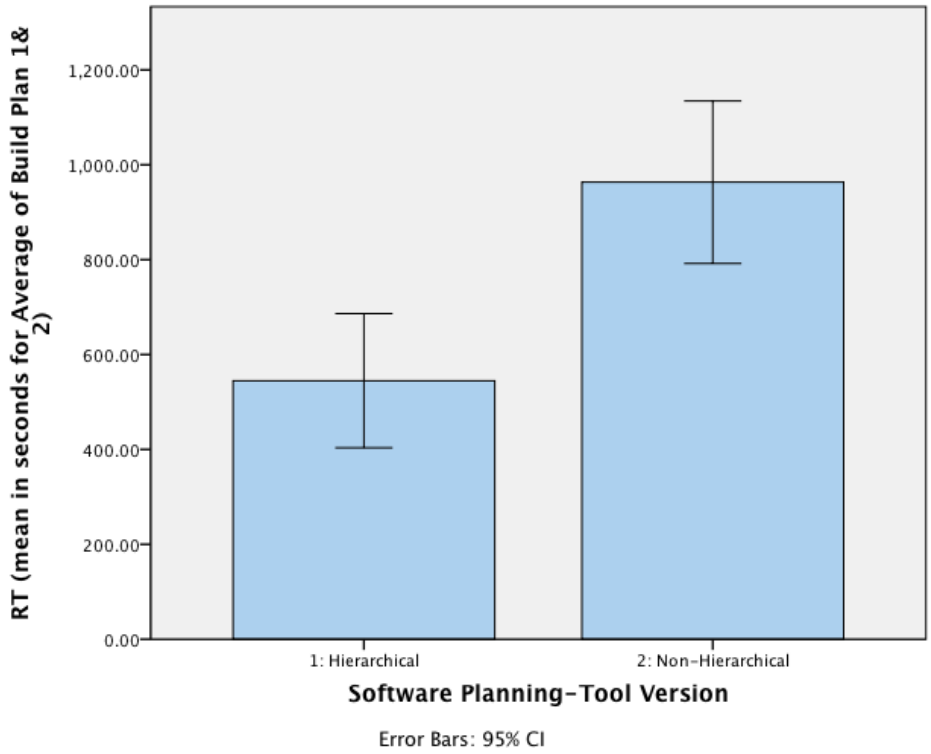

Figure 16. RT of Build Plan Task (Plan 1 and Plan 2) averaged across two plans that were built on each of the two versions of software planning tool

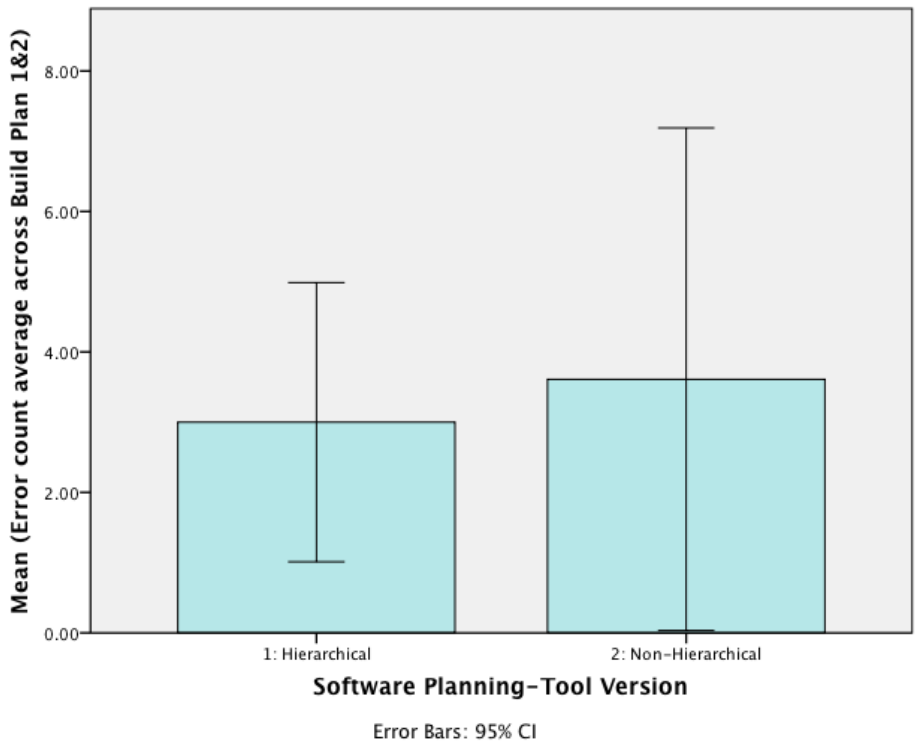

Figure 17. Error count for Build Plan Task (Plan 1 and Plan 2) averaged across two plans that were built on each of the two versions of software planning tool.

In summary, the participants in the Hierarchical Condition were significantly faster than their counterparts in the Non-Hierarchical Condition in building plans, as predicted. Users in the Hierarchical Condition also tended to have fewer errors in the final plans compared to the users in the Non-Hierarchical Condition, but this difference in errors was not statistically significant.

Error-Finding Task

Independent sample *t* tests were performed to determine the effect of the Software planning tool (Hierarchical and Non-Hierarchical) on users' performance in detecting errors in the provided erroneous plan. The performance measures tested were the RT for the entire task and the number of correctly detected errors in the plan.

There was no significant difference in RT for Hierarchical (M = 1003.40, SD = 434.89) and Non-Hierarchical (M = 928.3, SD = 297.13) Software planning tool Conditions for the Error-

Finding Task, $t(18) = .45$, $p = .657$ (see Figure 18). No significant difference was found in correctly detected errors for the Hierarchical ($M = 5.20$, $SD = 2.82$) and Non-Hierarchical ($M = 6.60$, $SD = 3.17$) Software planning tool Conditions, $t(18) = -1.04$, $p = .311$ (see Figure 19).

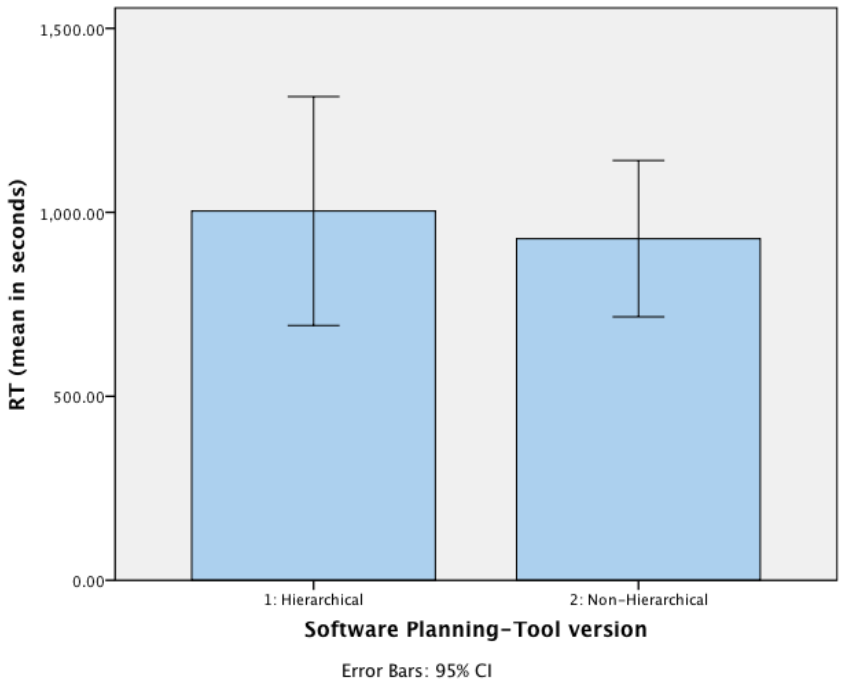

Figure 18. RT of Error-Finding Task across two versions of software planning tool.

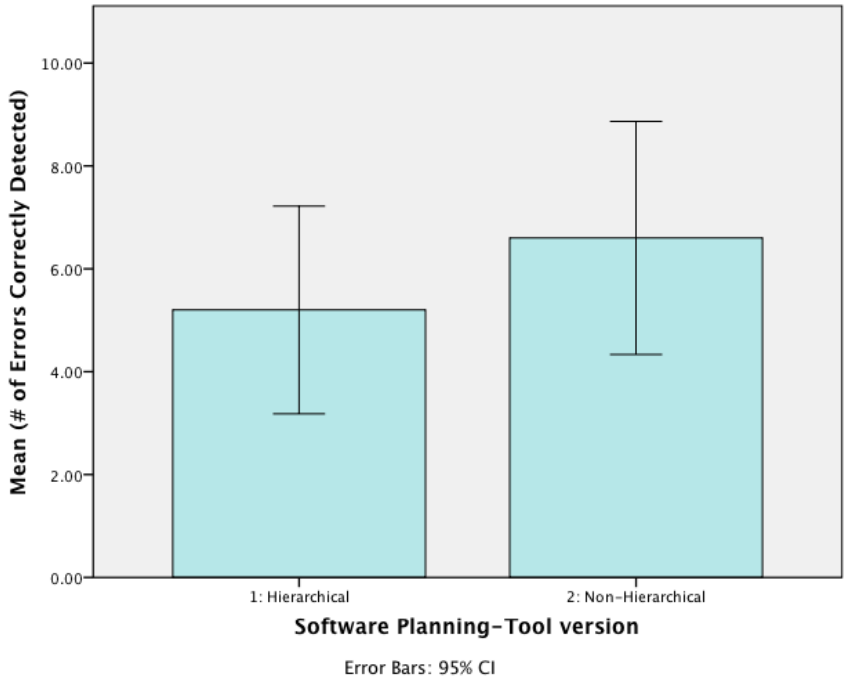

Figure 19. Mean of correctly detected errors in the Error- Finding task.

In summary, contrary to the prediction, the users in the Hierarchical Condition tended to take longer to report the errors than the users in the Non-Hierarchical Condition. A similar trend was also noted in the reporting of scripted errors, where users in the Non-Hierarchical Condition tended to report a higher count of correctly detected errors than their counterparts in the Hierarchical Condition. However, neither condition effect was statistically significance.

Debriefing

In the end of each session users were prompted to speak of their experience of interacting with the software planning tool. Overall most of the users stated that they had positive experience learning about a new domain and interacting with the software planning tool. However they also voiced the challenges they faced and it ranged from their overall experience of the study to finer details of their interaction experience with software planning tool. Users mentioned frequently that lack of enough training posed challenges in pacing up with software

planning tool usage proficiency. They further mentioned that while working on plans they felt lack of control in manipulating the events on timeline. Some users also found it challenging to switch modes while comparing plans.

Discussion

Overview

The purpose of the study was to investigate the effect of alternative interface designs for a software planning tool on user performance. The study compared two conditions, each using a different version of the software planning tool and users' performance was measured and compared. To measure the impact of the two versions, tasks were grouped as those using the planning tool and those assessing domain knowledge, which were performed after the use of the software planning tool. The primary goal of the study was to ascertain if any performance differences existed between users who worked on the two versions of the software planning tool. Performance was measured as time taken to complete the task and error counts in each task.

The secondary goal of the study was to explore whether there was an impact of the software planning tool version on the overall understanding of the relationships among plan elements in the ADCO domain after the participant gained experience from working on the software planning tool. To assess whether the software planning tool condition influenced learning, one task was presented before and one after using the software planning tool and we tested for interaction of software planning tool condition with before versus after software use as a factor.

As the Hierarchical version presented a closer visual match of a plan to that of the underlying schema of the plan elements, it was hypothesized that the study would show a positive advantage of Hierarchical representation over a Non-Hierarchical one. Furthermore, it was also hypothesized that this advantage would result in observable differences in a user's understanding of the temporal relationships of the elements of a plan and would be visible on tasks that were designed to measure this. We hypothesized that users in the Hierarchical

Condition would demonstrate superior performance than their counterparts in the Non-Hierarchical condition.

The next section discusses the users' performance across the different task types, categorized broadly as tasks performed using the software planning tool and tasks assessing domain understanding (that were performed without using the planning tool). Overall, though the study indicated the data to be trending toward the hypothesis, very few effects of software planning tool condition type were found to be statistically significant.

Edit-Time Task

For the Edit-Time Task, we predicted that participants in the Hierarchical Condition would demonstrate an advantage over their counterparts in the Non-Hierarchical Condition in editing of the elements of a plan (Actions and/or Activities). We also predicted an interaction of Software planning tool Conditions and Item Types.

Overall, the results for the Edit-Time Task were mixed and did not particularly favor one condition over the other. Though there were task types where the participants in the Hierarchical Condition did perform better than the ones in the Non-Hierarchical Condition and some did get close to statistical significance, the majority of the task type findings did not reach statistical significance.

In a few instances, the analyses comparing differences in performance for each item type across the two versions of the software planning tool did yield a marginal main effect of software planning tool version. For example, comparing individual Activity versus individual Action item-type, participants in the Hierarchical Condition were significantly faster than users in the Non-Hierarchical Condition in correctly editing an individual Activity than editing an individual Action. This aligned well with the prediction that a direct visual cue of Actions nested as a

component of a parent Activity would better enable the user to better operate at Action and Activity level in the Hierarchical version than in the Non-Hierarchical version. This prediction was reflected in the participants' performance in the Edit-Time Task.

Furthermore, for item type in Edit-Time Task comparing actions grouping Within and Between Activities, we hypothesized both a main effect and interaction effect. We hypothesized that overall Within Activity items would be easier (shorter RT and fewer errors) than Between Activities items for both the software planning tool condition. However, the difference between these two items types was predicted to be more pronounced for participants in the Hierarchical Condition than for the ones in the Non-Hierarchical Condition. This prediction was based on the expectation that the delineation of Actions as a sub-element of Activity in the Hierarchical version would make the selection of Actions more accurate and faster for each Within and Between spatial distribution. The pattern was weakly suggested in the data, but neither main nor interaction effects were significant.

Build-Plan Task

The Build-Plan Task was one of the two tasks that were performed on the software planning tool. This task entailed building a plan from the elements (Actions and Activities) in the template by putting the elements in the correct temporal sequence. For the Build-Plan Task, we predicted that users in the Hierarchical Condition would have a measurable advantage over the users in the Non-Hierarchical Condition. Such an effect could be attributed to experience gained by the explicit visual layout delineating the hierarchical organization of the elements of a plan as compared to the flat representation in the Non-Hierarchical version.

The findings did indicate a significant impact of software planning tool version on RT. The Build-Plan Task was the only task that showed a statistically significant impact of version

type on a performance metric, RT. For the error count, the difference across each version type wasn't statistically significant, but users in the Hierarchical Condition did have overall fewer errors compared to their counterparts in the Non-Hierarchical Condition.

Multiple-Choice Questions

For this task type, the general hypothesis stated that participants in each of the two conditions, Hierarchical and Non-Hierarchical, would improve from first presentation to the second presentation after they had worked with the software planning tool. Overall, there was no statistically significant difference in RT before and after working with the software planning tool across the two conditions. However, the users tended to respond faster the second time after using the software planning tool. It was also of interest to note that users in the Hierarchical Condition, who were slightly slower than their counterparts in answering the questions before using the software planning tool, sped up for the second test. But none of these improvements was large enough to be statistically significant.

To summarize, the findings of the multiple-choice questions may align with the prediction that using a software planning tool would have a positive impact on a user's understanding of the relationship of an action sequence within an Activity. However, any such impact was not strong enough to be statistically significant.

Error-Finding Task

The Error-Finding Task was performed after the Build-Plan Task and was performed without using the software planning tool. Users were provided with a plan that included 12 scripted errors, and they were instructed to report the errors. We predicted that users in the Hierarchical Condition would demonstrate better performance, as measured by lower RT, and

that they would correctly report more errors than their counterparts in the Non-Hierarchical condition.

There was no statistically significant impact of the software planning tool version on RT or on error reports. Interestingly, the trend was opposite to that predicted; users in the Non-Hierarchical were faster and reported a higher error count than their counterparts in the Hierarchical Condition.

Limitations of the Current Study and Scope for Future Research

When introduced to the experiment, the users were all new to the ADCO domain. To acquaint them with the ADCO domain fundamentals, each user was provided with an ADCO domain-training document to read through. Considering the nature of our users profiles (aviation students aspiring to work in aerospace), the ADCO training intrigued many users and some wanted to spend more time than allocated to get a better insight into ADCO. Such user preference resulted in instances where the user session ran longer than anticipated. Some users referred to the ADCO training and mentioned that though interesting, it was an information overload. At the end of the experiment during the debriefing sessions, some users reported that the long experimental session resulted in fatigue towards the end of the session and might have impacted their performance. In practice, an ADCO officer would already have gone through the process of learning about the domain, so performing tasks using a novel software planning tool would be faster, with a lower workload from learning and retaining new information.

To have as close a surrogate of ADCO officers as feasible for the experiment, all the users were chosen from a pool of students in the Aviation Engineering Department at San José State University. Furthermore, to ensure an equivalent user representation across the two condition types, users were allocated to each condition based on their performance on the ADCO

training questionnaire. Despite such measures to balance the conditions, three of the 10 participants in the Hierarchical Condition had performance data that was much below the average performance of the rest of the sample users across both the conditions. Review of their video recordings showed that these users only operated at the level of Activity, thus producing incorrect response for 5 of the 6 item types. This resulted in only four overall correct responses of the 24 tasks in all in Edit-Time Task. Two of the three users were seen to be expanding Activities and individually selecting Actions in a few instances in the screen recordings, but these users didn't edit the selected Actions and chose to edit the parent Activity instead at all the instances. While analyzing the possible reasons for this user approach of only working at Activity level, the plausible factors included: inadequate attention paid by the user to the presented task or insufficient time for hands-on work with the software planning tool resulting in poor understanding of the functionality of the tool. Since two of the three users were seen to be clicking and selecting the Actions in plan, it could be surmised that they were aware of the Action level in the hierarchy but not too well versed in editing the Actions. This could have resulted due to lack of adequate training with the tool. It is also notable that most of the users showed some learning effect, tending to err more in the first of the six blocks in the Edit-Time Task and eventually improving over the next blocks. In contrast, these three users never seemed to pause throughout the 24 tasks and went ahead with consistent high speed through the task. As expected, their RT was much less than the rest of the users in the Hierarchical Condition. This also reflects on the three users' rushed approach while performing the task.

Since the root cause of such extreme performance is unclear, a plausible intervention would be the software planning tool training. By letting the user have more time to work hands-on with the software planning tool, we might ensure improved understanding and reduced error.

Another plausible approach to improve robustness of the experiment would be to have a larger sample size. A larger sample size would have provided greater statistical power with an overall more representative data.

Follow up research should improve the current study method for domain and tool training. A more efficient training approach could involve automating the training for domain and software planning tool to ensure the standard coverage and mastery of material.

Conclusion

The purpose of this thesis was to determine the impact of the Hierarchical and Non-Hierarchical versions of the software planning tool on user's performance. We found little effect of the version of the software planning tool. The one task where users in the Hierarchical version scored significantly better than their counterparts in the Non-Hierarchical version was the Build Plans Task. Besides the Build Plans Task, no other remaining task, such as the Edit-Time Task or the Error-Finding Task, indicated significant impact of the software planning tool version type on performance metric. However because working in a niche and specialized domain as ADCO's requires exhaustive training, it was very encouraging to see that the majority of users could comply well with the domain and tool training and addressed each task with ease.

The data in the study did trend toward the hypothesis of advantage of Hierarchical information organization over Non-Hierarchical organization in planning tool context. But the trend was not strong enough to be statistically proven. Based on the findings of the experimental study there were few plausible directions that could have led to higher confidence in our estimates with less uncertainly and greater precision. Larger sample size could have helped us handle the loss of three data points of a sample size of 10 in Hierarchical condition for Edit Time

task. Further better software planning tool training might have also helped with improved user confidence resulting in better precision while working on software planning tool.

REFERENCES

Bennett, J. (1996). What Events Are. In R. Casati & A. C. Varzi (Eds.), *Events* (pp. 136-152). Aldershot, England: Dartmouth.

Beyer, H., and Holtzblatt, K. *Contextual Design: Defining Customer- Centered Systems.* San Francisco, CA: Morgan Kaufmann (1998).

Biederman, I. (1987). Recognition–by-components: A theory of human image understanding. *Psychological review*, 94, 114-118.

Billman, D., Arsintescu,L., Feary, M., Lee, J., Smith, A., & Tiwary, R. (2011).The benefits of matching the domain structure for planning software: The right stuff. *In Proc. CHI 2011, ACM Press (2011).*

Boltz, M. G. (1992b). The Remembering of Auditory Event Durations. *Journal of Experimental Psychology-Learning Memory and Cognition*, 18, 935-958.

Burns, C. M., & Hajdukiewicz, J. R. (2004). *Ecological interface design*. Boca Raton, FL: CRC Press.

Butler, K. A, Jiajie, J., Esposito, C., Ali Bahrami, A. Ron Hebron, R., & David Kieras. Work-Centered Design: A Case Study of a Mixed-Initiative Scheduler. In *Proc. CHI 2007*, ACM Press (2007).

Butler & Zang (2009). Design models for interactive problem-solving context & ontology, representation & routines. In *Proc*. CHI 2009, ACM Press pp. 4214-4320.

Diaper, D. & Stanton, N.A. (2004). *The Handbook of task Analysis for Human-Computer Interaction*. LErlbaum Associates, Mahway, NJ.

Hanson, C., & Hirst, W. (1989*)*. On the representation of events: A study of orientation, recall, and recognition. *Journal of Experimental Psychology*: General, 118, 136-147.

Johnson, H., & Johnson, P. (1991). Task knowledge structures: Psychological basis and integration into system design. *Acta Psychologica*, 78, 3-26.

Morris, M., & Steedman, M. (1998). Converging operations on a basic level in event taxonomies. *Memory & Cognition*, 18, 406-420.

Rasmuseen, J., Pejtersen, A.M., & Goodstein, L. P. (1994). *Cognitive systems engineering*. New York: Wiley.

Schank, R. C., & Abelson, R. P. (1977). *Scripts, plans, goals, and understanding: an inquiry into human knowledge structures*. Hillsdale, N. J.: L. Erlbaum Associates.

Tversky, B. (1990). Where partonomies and taxonomies meet. In S. L. Tasohatzidis (Ed.), *Meanings and prototypes: studies in linguistic categorization* (pp. 334-344). London: Routledge.

Tversky, B., & Hemenway, K. (1984). Objects, parts, and categories. *Journal of Experimental Psychology*: general, 113, 169-195.

Wortman, M., P. (1975). Long-Term Retention of Information as a function of its Organization. *Journal of experimental Psychology: Human Learning and memory*, 576-577.

Vallacher, R. R., & Wegner, D. M. (1987). What do people think they're doing? Action identification and human behavior. *Psychological Review*, 94, 3-16.

Vincente. K. (1999). *Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer-based Work*. Erlbaum Associates, Mahway, NJ.

Zacks, J. M. T., B. (2001). Event structure in perception and conception. *Psychological Bulletin*, 127, 3-21.

Zacks, J. M., Tversky, B., & Iyer,G. (2001). Perceiving, Remembering, and Communicating Structure in Events. *Human Factors: The Journal of the Human Factors and Ergonomics Society, 49*(6), 1115-1131.

Zhang J. (1996). A representational analysis of relational information displays. *International Journal of Human Computer Studies*, 45, 59-74.

Appendix A: San José State University IRB Approval

SAN JOSÉ STATE
UNIVERSITY

Division of Academic Affairs
Associate Vice President
Graduate Studies & Research
www.sjsu.edu/gradstudies

One Washington Square
San José, California 95192-0025
Voice: 408-924-2427
Fax: 408-924-2612

www.sjsu.edu

To: Rachna Tiwary

From: Pamela Stacks, Ph.D.
Associate Vice President
Graduate Studies and Research

Date: March 3, 2011

The Human Subjects-Institutional Review Board has approved your request to use human subjects in the study entitled:

"A comparison of alternative interface designs for planning softwares"

This approval is contingent upon the subjects participating in your research project being appropriately protected from risk. This includes the protection of the confidentiality of the subjects' identity when they participate in your research project, and with regard to all data that may be collected from the subjects. The approval includes continued monitoring of your research by the Board to assure that the subjects are being adequately and properly protected from such risks. If at any time a subject becomes injured or complains of injury, you must notify Dr. Pamela Stacks, Ph.D. immediately. Injury includes but is not limited to bodily harm, psychological trauma, and release of potentially damaging personal information. This approval for the human subject's portion of your project is in effect for one year, and data collection beyond March 3, 2012 requires an extension request.

Please also be advised that all subjects need to be fully informed and aware that their participation in your research project is voluntary, and that he or she may withdraw from the project at any time. Further, a subject's participation, refusal to participate, or withdrawal will not affect any services that the subject is receiving or will receive at the institution in which the research is being conducted.

If you have any questions, please contact me at (408) 924-2427.

Protocol #S1102021

cc. Kevin Jordan 0120

Appendix B: San José State University Informed Consent

San José State
UNIVERSITY

Department of Psychology
DMH 157
One Washington Square
San Jose, CA 95192-1020
Voice: 408-924-5600
Fax: 408-924-5605
www.psych.sjsu.edu

Agreement to Participate in Research

Responsible Investigator: Rachna Tiwary, student, Human Factors and Ergonomics, Industrial and Systems Engineering, SJSU

Title of Study:

Effects of hierarchical vs. non–hierarchical representation of software application interface on performance of user's domain specific task.

1. You have been asked to participate in a research study that investigates the effects of change in application interface on user performance and their understanding of domain knowledge. The application investigated in this study is used for planning the movement and orientation of International Space Station (ISS) by a niche group of officers known as ADCO (Attitude Determination and Control Operator). You will be asked to perform tasks that will be similar to task performed by ADCO on this application.

2. You will work on one of the two versions of the application. Each version will vary only in the representation of information on the application's interface. You will be provided with instructions to perform tasks on the planning application. You will be performing this task on a computer using mouse and keyboard. You will see both the application and the instruction on the monitor. You will be asked to make edits to existing plans on the application.

3. The study will be conducted in AIDE (Automation Interaction Design and Evaluation) group lab Facilities at NASA Ames. Devices that will be used are a 32" monitor, a digital mouse and a keyboard. Morae, a software application by Techsmith, used for usability testing and user experience research, will be used to record the user's screen during the study session.

4. There will not be any risks present in the study outside of what are present in daily life.

5. There will not be any direct benefit in this study although an indirect benefit can be a better insight and knowledge of ADCO's domain and ISS planning.

6. Although the results of this study may be published, no information that could identify you will be included.

7. The participant will get monetary compensation for their participation. The study is expected to run for approximately for 4 hours. Compensation will be proportional to the time spared by the participant to complete the study. Participant will be paid at the rate of $12.73 per hour. Estimated compensation will be approximately $ 50.92 for the entire study.

8. Questions about this research may be addressed to Rachna Tiwary, phone number (408)368-2560 ,email : ractiw@gamil.com. Complaints about the research may be presented to Dr Kevin Jordan, Professor of Psychology, San Jose State University, phone number (650)604-6018. Questions about a research subjects' rights, or research-related injury may be presented to Pamela Stacks, Ph.D., Associate Vice President, Graduate Studies and Research, at (408) 924-2427.

9. No service of any kind, to which you are otherwise entitled, will be lost or jeopardized if you choose not to participate in the study.

San José State
UNIVERSITY

Department of Psychology
DMH 157
One Washington Square
San Jose, CA 95192-1020
Voice: 408-924-5600
Fax: 408-924-5605
www.psych.sjsu.edu

10. Your consent is being given voluntarily. You may refuse to participate in the entire study or in any part of the study. If you decide to participate in the study, you are free to withdraw at any time without any negative effect on your relations with San Jose State University or NASA Ames.

11. At the time that you sign this consent form, you will receive a copy of it for your records, signed and dated by the investigator.

- **The signature of a subject on this document indicates agreement to participate in the study.**
- **The signature of a researcher on this document indicates agreement to include the above named subject in the research and attestation that the subject has been fully informed of his or her rights.**

Participant's Signature	Date
Investigator's Signature	Date

Appendix C: Build Plan Task instruction document (Hierarchical Condition)

Task 1: 1st Plan

Build a plan comprising of 4 activities namely **Thruster Disable, Docking, one more Thruster Disable and Reboos**t (in that Oder):

Following are the action sequence in the build plan

Thruster Disable Activity made up of

- Config Mng
- Mom Mng

Docking Activity made up of

- HO to RUS
- Mnvr to Duty Att
- Mnvr to dock
- Free drift
- Mnvr to TEA
- HO to US

Reboost Activity is made up of

- HO to RUS
- Mnvr to Duty Att
- Mnvr to Reboost
- Mnvr to TEA
- HO to US

Following are the instructions to guide you build the plan:

1. Time need not be too precise (you are expected to work on drafting the plan right and space events in it as instructed. Exact precision on individual event's schedule isn't required).

2. To drag the events (Activity), go to the 'Table' mode and drag relevant actions from the template. To edit the time and 'spacing' between event switch to the 'Timeline' mode.

3. Space out the Activities so that the 1st three Activities are occurring on first day of the plan and last one Activity occurs on Day 2 of the plan.

4. Place the 'Docking' Activity 1 hour apart from the 1st Thruster Disable (1 hour apart would mean the **difference between** the **End Time of the Thruster Disable** and **Start Time of Docking** is 1 hour)

5. Space the 2nd Thruster Disable 4 hours apart from the 'Docking' Activity (4 hours apart implies that the **difference between** the **End Time of the Docking** and **Start Time of 2nd Thruster Disable** is 4 hours)

Task 2: 1st Plan

Build a plan comprising of 4 activities namely Thruster Disable, Undocking, one more Thruster Disable and Relocate (in that Oder):

Following are the action sequence in the build plan

Thruster Disable Activity made up of

- Config mng
- Mom Mng

Undocking Activity is made up of

- HO to RUS
- Mnvr to Duty Att
- Mnvr to Undock
- Free drift
- Mnvr to TEA
- HO to US

Relocate activity is made up of

- HO to RUS
- Mnvr to Undock
- Free drift
- Mnvr to Dock
- Current Att
- Free drift
- Mnvr to Duty Att
- HO to US

Following are the instructions to guide you build the plan:

1. Time need not be too precise (you need to work on drafting the plan right and space events in it as instructed and not focus too much on individual event's schedule).

2. To Drag the events (Actions), so to the 'Table' mode and drag relevant actions from the template. To edit the time and 'spacing' between event switch to the 'Timeline' mode.

3. Space out the Activities so that the 1st two Activities are occurring on first day of the plan and last two Activities occurs on Day 2 of the plan.

4. Place the 'Undocking' Activity 2 hours apart from the 1st Thruster Disable (2 hour apart would mean the **difference between** the **End Time of the Thruster Disable** and **Start Time of Undocking** is 2 hour)

5. In the Undocking activity ,space out the 'Free drift' to be 1hour 30 mins apart from the HO to RUS. This should be so done that the spacing between 'Free Drift ' and its following two actions (Mnvr to TEA and HO to US) remains unchanged.

6. In the Relocate activity ,space out the 'Current Att' to be 2 hours apart from the Mnvr to Undock .This should be so done that the spacing between 'Current Att' and its following 3 actions (Free Drift, Mnvr to Duty att and HO to US) remains unchanged.

Appendix D : Build Plan Task instruction document (Non Hierarchical Condition)

Task 1: 1st Plan

Build a plan comprising of 4 activities namely **Thruster Disable, Docking,** one more **Thruster Disable and Reboost (in that Oder):**

Following are the action sequence in the build plan

- Config mng
- Mom Mng

Makes up **Thruster Disable Activity**

- HO to RUS
- Mnvr to Duty Att
- Mnvr to Dock
- Free drift
- Mnvr to TEA
- HO to US

Makes up **Docking Activity**

- HO to RUS
- Mnvr to Duty Att
- Mnvr to Reboost
- Mnvr to TEA
- HO to US

Makes up **Reboost Activity**

Following are the instructions to guide you build the plan:

6. Time need not be too precise (you need to work on drafting the plan right and space events in it as instructed and not focus too much on individual event's schedule).

7. To Drag the events (Actions), go to the 'Table' mode and drag relevant actions from the template. To edit the time and 'spacing' between event switch to the 'Timeline' mode.

8. Space out the Activities so that the 1st three Activities are occurring on first day of the plan and last one Activity occurs on Day 2 of the plan.

9. Place the ‘Docking’ Activity 1 hour apart from the 1st Thruster Disable (1 hour apart would mean the **difference between** the **End Time of the Thruster Disable** and **Start Time of Docking** is 1 hour)

10. Space the 2nd Thruster Disable 4 hours apart from the ‘Docking’ Activity (4 hours apart implies that the **difference between** the **End Time of the Docking** and **Start Time of 2nd Thruster Disable** is 4 hours)

Task 2: 2nd Plan

Build a plan comprising of 4 activities namely Thruster Disable, Undocking, one more Thruster Disable and Reloctae (in that Oder):

Following are the action sequence in the build plan

- Config Mng
- Mom Mng

Makes up **Thruster Disable Activity**

- HO to RUS
- Mnvr to Duty Att
- Mnvr to Undock
- Free drift
- Mnvr to TEA
- HO to US

Makes up **Undocking**

- HO to RUS
- Mnvr to Undock
- Free drift
- Mnvr to Dock
- Current Att
- Free drift
- Mnvr to Duty Att
- HO to US

Makes up **Relocate Activity**

7. Time need not be too precise (you need to work on drafting the plan right and space events in it as instructed and not focus too much on individual event's schedule).

8. To Drag the events (Actions), so to the 'Table' mode and drag relevant actions from the template. To edit the time and 'spacing' between event switch to the 'Timeline' mode.

9. Space out the Activities so that the 1st two Activities are occurring on first day of the plan and last two Activities occurs on Day 2 of the plan.

10. Place the 'Undocking' Activity 2 hours apart from the 1st Thruster Disable (2 hour apart would mean the **difference between** the **End Time of the Thruster Disable** and **Start Time of Undocking** is 2 hour)

11. In the Undocking activity ,space out the 'Free drift' to be 1hour 30 mins apart from the HO to RUS.This should be so done that the spacing between 'Free Drift ' and its following two actions (Mnvr to TEA and HO to US) remains unchanged.

12. In the Relocate activity ,space out the 'Current Att' to be 2 hours apart from the Mnvr to Undock .This should be so done that the spacing between 'Current Att' and its following 3 actions (Free Drift, Mnvr to Duty att and HO to US) remains unchanged.

Appendix E: Error Finding Task Template

Error Finding Task Plan: "FindError.Plan"

This plan shows a sequence of activities, with errors. Look over the plan displayed, and identify as many errors as you can. Errors might include missing actions, extra actions, incorrect events (wrong type), or wrong display of an action as a point versus interval event. You might see other, uncategorized errors or think about errors differently.

Briefly describe each error you see, using the form below

	Missing action	Extra action	Incorrect action	Point/interval	Other (or uncategorized)
Thruster Disable					
Reboost					
Thruster Disable					
Russian EVA					
Thruster Disable					
Thruster Test (RS Master)					

Appendix F: Debriefing Questions

Software questions:

1) What aspect(s) of this planning software prototype did you like the most and why?

2) What aspect(s) of this software did you dislike or find annoying/frustrating/or confusing and why?

3) Any general comments of your experience about the software prototype, its (ease or difficulty) of use?

4) Any suggestion about the prototype's interface design improvements that you think would have made it easier or more intuitive to use/work with?

Task questions:

5) What aspects or parts of the task were most fun or interesting?

6) What aspects or parts of the task did you feel were particularly difficult or confusing?

Strategy questions:

7) Ask about usual strategy and any variations for viewing and changing info. For example, if participant wanted to find the time when a particular action started, where exactly did he or she look?